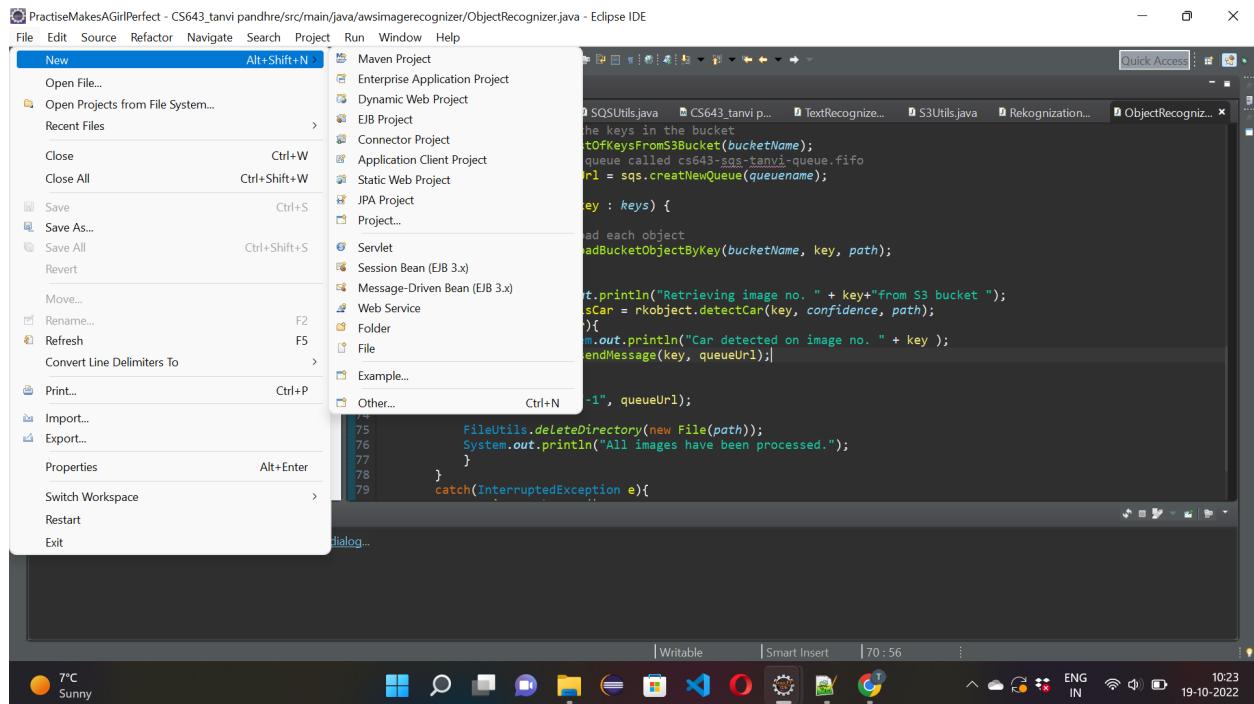
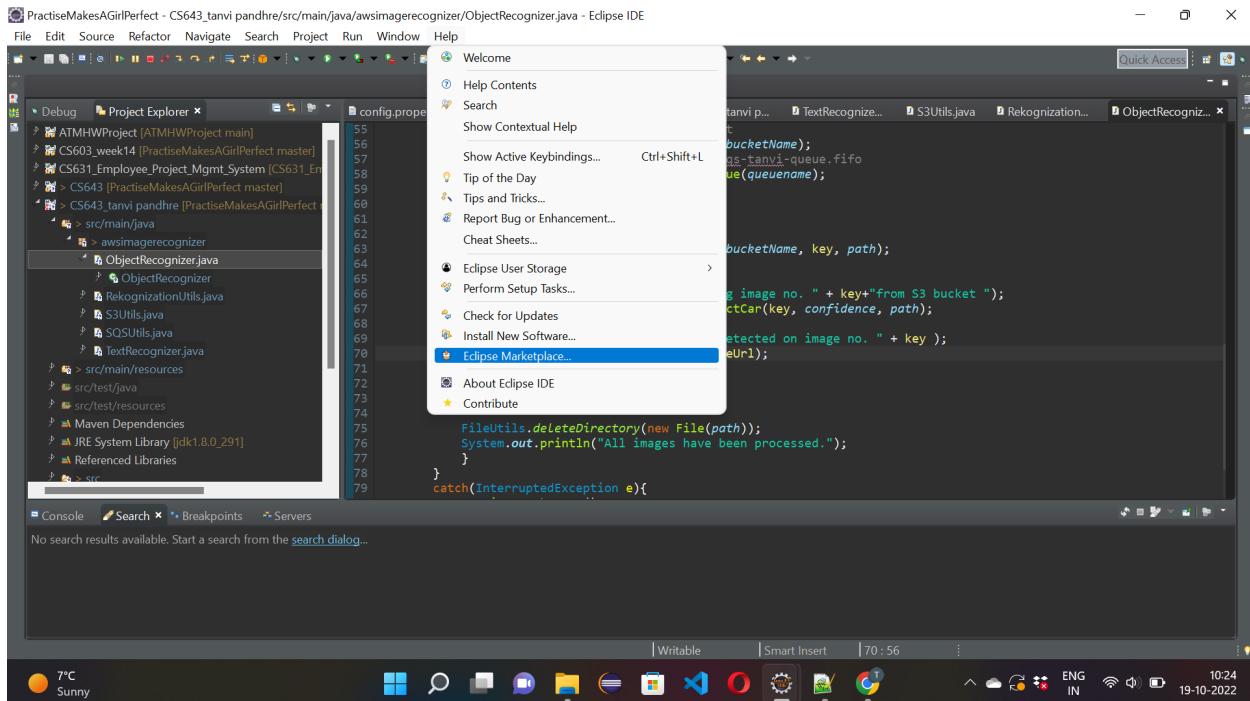


## #Installation details.

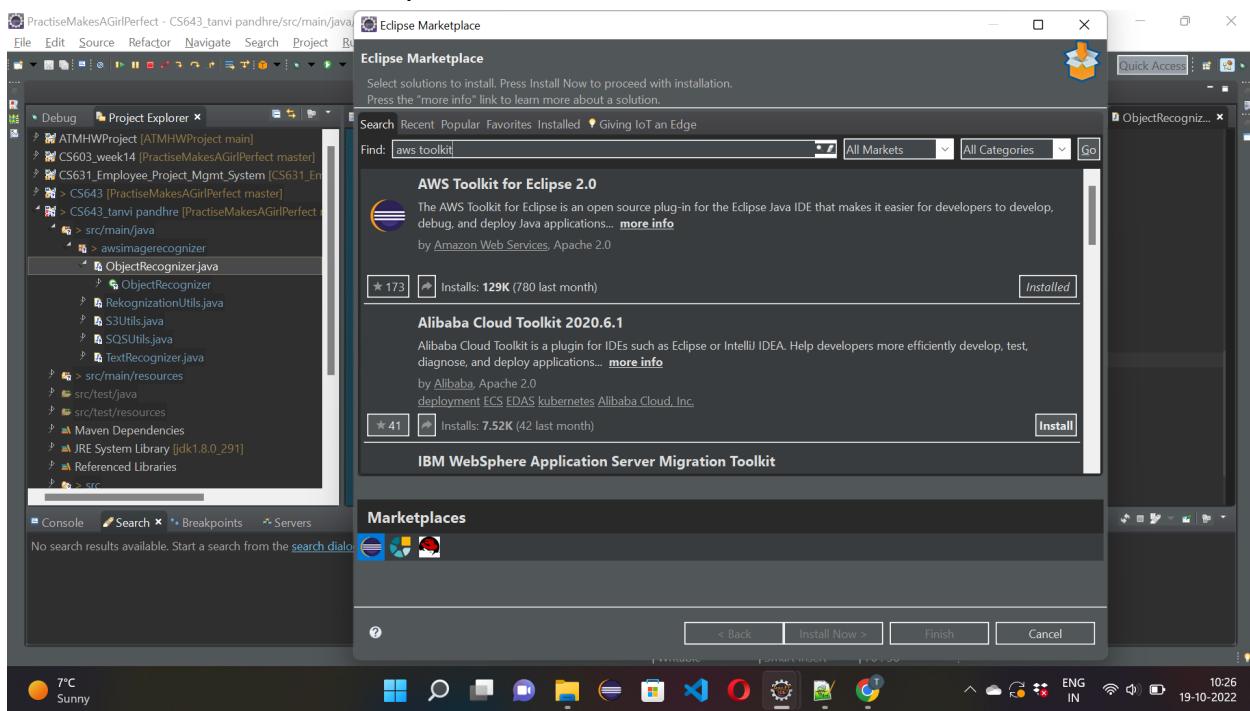
1. Use eclipse IDE.
  2. Create a maven project called CS643\_tanvi pandhare. By choosing File -> New -> Maven Project.



3. Install aws toolkit in eclipse. By choosing Help -> Eclipse Marketplace. Search aws toolkit to install the software.



When it asks for aws access key. Provide from aws details of the learner's lab aws module.



# File information - Packages, Java files, properties file, xml.

1. config.properties file -

Mention bucketname, confidenceLevel, pathOfImages, resultantPath, nameOfSqsQueue.

2. Create package awsimagerecognizerUtils for all the utility methods.
3. RekognitionUtils.java - Methods related to aws recognition are written in this class.
  - detectObject() - This method will accept image, confidence level and path as parameters. In this method AmazonRekognition agent is created that will detect the object label 'Car' with the required confidence level.
  - detectText() - This method will accept image and path as parameters. In this method AmazonRekognition agent is created that will detect text from the image that is passed.
4. S3Utils.java - Methods related to aws s3 bucket are written in this class. It provides a simple interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web.
  - S3Utils() - In this constructor s3Client is created with the credentials and region is created.
  - isBucket() - checks if bucket exists with the passed bucketName using the s3Client's function doesBucketExistV2()
  - listOfKeysFromS3Bucket() - it returns a list of keys from the s3 bucket. using the s3Client's function listObjectsV2().
  - downloadBucketObjectByKey() - This method will accept bucket name, key name and path as parameters and we can download the particular image from the given bucket.
5. SQSUtils.java - Methods related to aws Simple Queue Service are written in this class.
  - getQueueUrl() - It generates sqs queue url from the passed queue name
  - creatNewQueue() - This method will create a new fifo queue with a certain retention period. Will also check if the queue already exists or not. Create one if not
  - sendMessage() - It will generate a random unique identifier uuid. It will generate a message to be sent to the sqs queue.
  - receiveMessage() - It will receive all the messages passing through the sqs queue.
  - deleteMessage() - it will delete message passing through the sqs queue
  - deleteQueue() - it will delete the entire queue whose url is passed as parameter.
6. Create package awsimagerecognizer for the image and text recognizer files.
7. ObjectRecognizer.java - This file is for Instance a to detect object Car and send the indexes to sqs queue.
  - Instantiate all the above utility file objects to use its methods and load the config property file.
  - First it checks if the bucket exists. If it does exist then all the keys from the s3 bucket have to be retrieved.

- A new queue is created.
- Each object is loaded from the s3 bucket.
- Once we get the object it is checked whether the object is a car and send its index key to the sqs queue.
- Once instance A terminates its image processing, it adds index -1 to the queue to signal to instance B that no more indexes will come.

8. TextRecognizer.java -

- Instantiate all the above utility file objects to use tits methods and load the config property file.
- check if S3 bucket exists
- Each message received from sqs is iterated.
- New file output.txt is created.
- All the images from the s3 bucket are downloaded whose index is present in the sqs messages.
- Text of that particular image is detected from car image
- Hashmap is created that will store text containing index and text that is generated. This is stored in output.txt file.

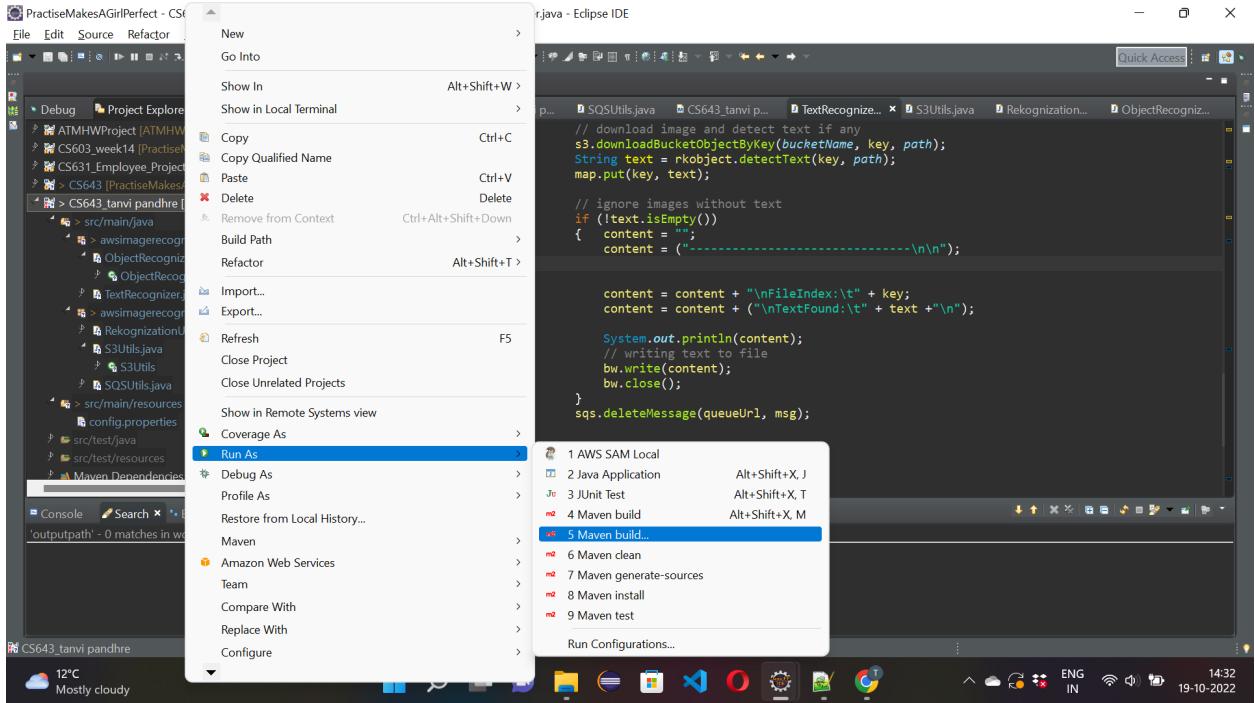
9. Pom.xml

In the tag mainClass - package name and class Name is mentioned

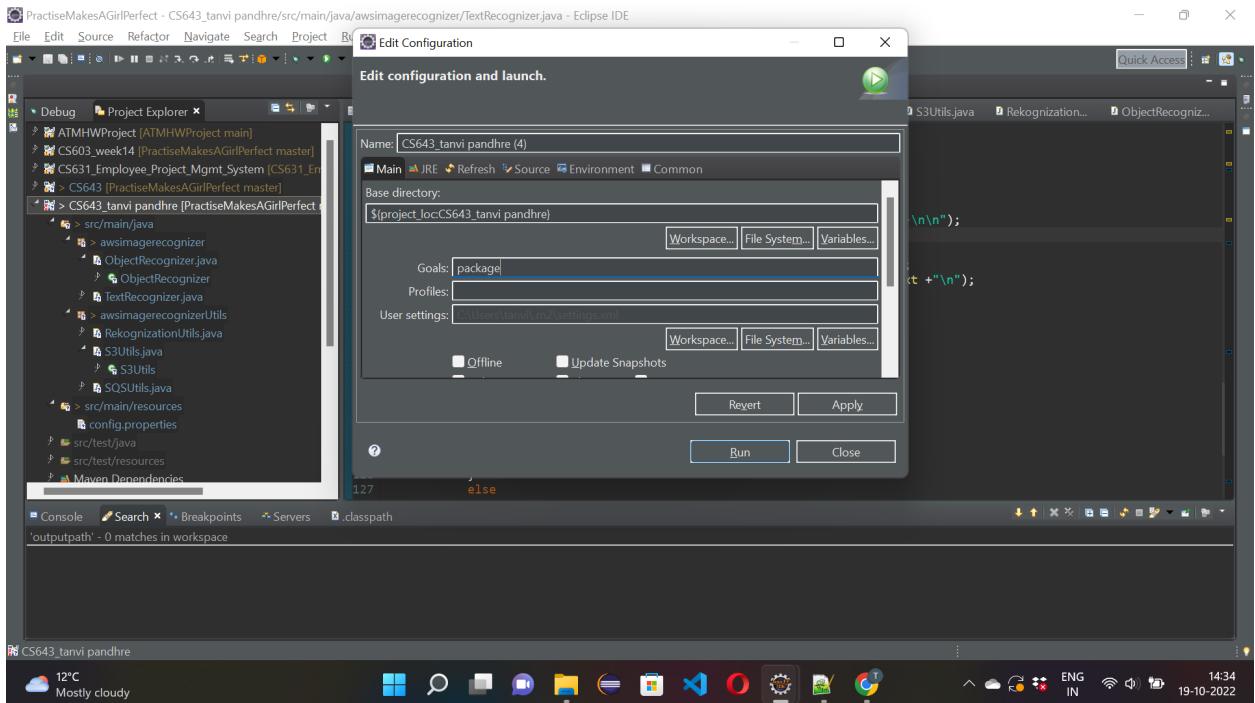
In the tag finalname name of the jar is mentioned.

## Instruction to use:

1. Create jars by doing Maven build.



## 2. In goals type package.



## 3. Jars should get successfully created in the target location. You Can check console.

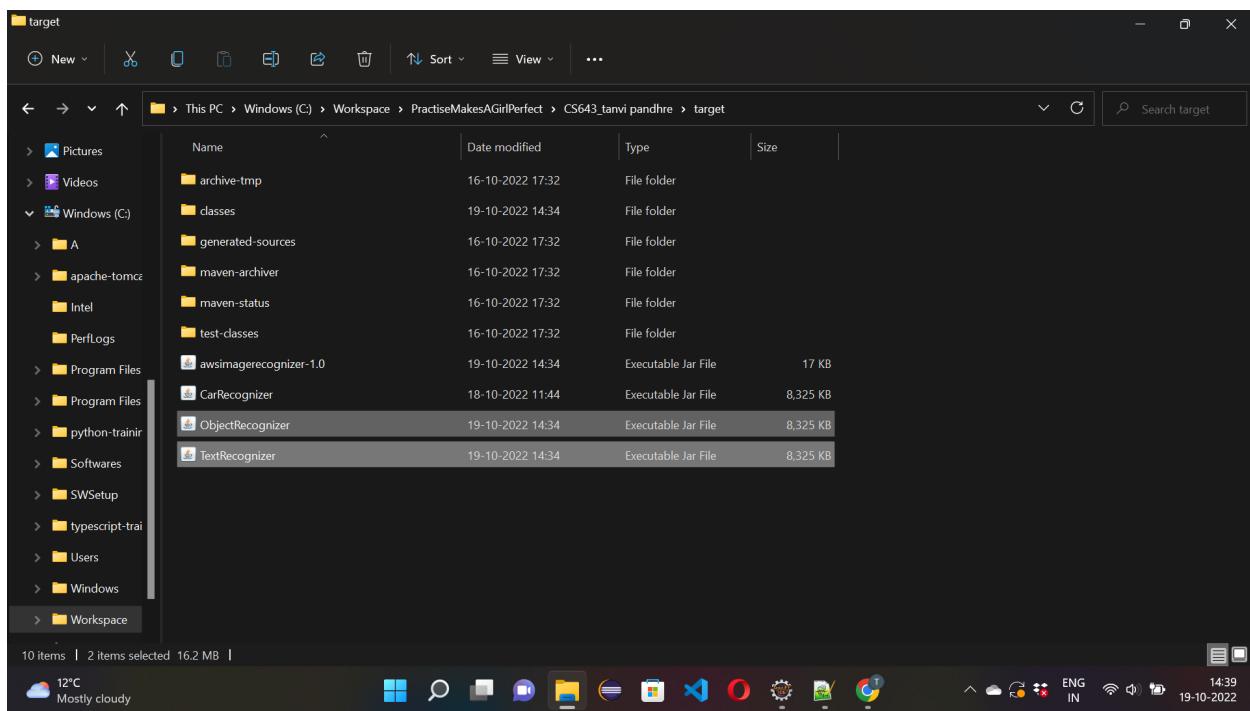
The screenshot shows the Eclipse IDE interface. In the center, there's a code editor window displaying Java code for a class named TextRecognizer. Below it is a terminal window showing the output of a Maven build process. The terminal output includes several INFO messages from various Maven plugins and configuration options. At the bottom, the Windows taskbar shows the date and time as 19-10-2022 14:35.

```

PractiseMakesAGirlPerfect - CS643_tanvi pandhare/src/main/java/awsimagerecognizer/TextRecognizer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer Debug config.property... CS643_tanvi p... SQSUtils.java CS643.tanvi p... TextRecognize... S3Utils.java Rekognition... ObjectRecogniz...
ATMHWProject [ATMHWProject main]
CS603_week14 [PractiseMakesAGirlPerfect master]
103 String text = rkoject.detectText(key, path);
104 map.put(key, text);
105 // ignore image without text
106

Console Search Breakpoints Servers classpath
<terminated> CS643_tanvi pandhare [4] [Maven Build] C:\Program Files\Java\jdk1.8.0_291\bin\javaw.exe (19-Oct-2022, 2:34:47 pm)
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ awsimagerecognizer ---
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ awsimagerecognizer ---
[INFO] Building jar: C:\Workspace\PractiseMakesAGirlPerfect\CS643_tanvi pandhare\target\awsimagerecognizer-1.0.jar
[INFO] --- maven-assembly-plugin:2.6:single (car-recognizer-assembly) @ awsimagerecognizer ---
[INFO] Building jar: C:\Workspace\PractiseMakesAGirlPerfect\CS643_tanvi pandhare\target\ObjectRecognizer.jar
[WARN] Configuration options: 'appendAssemblyId' is set to false, and 'classifier' is missing.
Instead of attaching the assembly file: C:\Workspace\PractiseMakesAGirlPerfect\CS643_tanvi pandhare\target\ObjectRecognizer.jar, it will become the file for main project
NOTE: If multiple descriptors or descriptor-formats are provided for this project, the value of this file will be non-deterministic!
[WARN] Replacing pre-existing project main-artifact file: C:\Workspace\PractiseMakesAGirlPerfect\CS643_tanvi pandhare\target\awsimagerecognizer-1.0.jar
with assembly file: C:\Workspace\PractiseMakesAGirlPerfect\CS643_tanvi pandhare\target\ObjectRecognizer.jar
[INFO]
[INFO] --- maven-assembly-plugin:2.6:single (text-recognizer-assembly) @ awsimagerecognizer ---
[INFO] Building jar: C:\Workspace\PractiseMakesAGirlPerfect\CS643_tanvi pandhare\target\TextRecognizer.jar
[WARN] Configuration options: 'appendAssemblyId' is set to false, and 'classifier' is missing.
Instead of attaching the assembly file: C:\Workspace\PractiseMakesAGirlPerfect\CS643_tanvi pandhare\target\TextRecognizer.jar, it will become the file for main project
NOTE: If multiple descriptors or descriptor-formats are provided for this project, the value of this file will be non-deterministic!
[WARN] Replacing pre-existing project main-artifact file: C:\Workspace\PractiseMakesAGirlPerfect\CS643_tanvi pandhare\target\ObjectRecognizer.jar
with assembly file: C:\Workspace\PractiseMakesAGirlPerfect\CS643_tanvi pandhare\target\TextRecognizer.jar
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 8.066 s
[INFO] Finished at: 2022-10-19T14:34:56+04:00
[INFO] -----

```



### ### Setup cloud Environment:

1. To create new EC2 instances go on aws console. Click on launch instances.
2. Select Amazon Linux Amazon Linux 2 Kernel 5.10 AMI free tier.

3. Keep instance type default - t2 tier.
4. Create new key-pair in the .pem format as tanvi\_keys\_instances2.pem
5. Allow SSH, HTTP, HTTPS traffic.
6. In security group chose MYIP - 128.235.85.0/32
7. Launch instances and you can see them running.

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table of instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
InstanceA	i-Off573521b1baef2f	Running	t2.micro	2/2 checks passed	No alarms
InstanceB	i-01ca48b0379f7491d	Running	t2.micro	2/2 checks passed	No alarms

A modal window titled "Select an instance" is open at the bottom, indicating the user is about to choose one of the listed instances.

8. Create a folder cs643 in AWS learners lab. Keep tanvi\_keys\_instances2.pem file in that folder.
9. Connect the EC2 command using the following commands.  
 chmod 400 tanvi\_keys\_instances2.pem  
 ssh -i "tanvi\_keys\_instances2.pem"  
 ec2-user@ec2-100-26-106-230.compute-1.amazonaws.com

Instance A is running

The screenshot shows the AWS Academy Learner Lab interface. On the left is a sidebar with navigation links: Home, Modules, Discussions, Account, Dashboard, Courses, Calendar, Inbox, History, Help, and a back arrow. The main area has tabs for AWS, Start Lab, End Lab, AWS Details, Readme, and Reset. It shows a usage of \$1 of \$100 and a region selector set to EN-US. The central part displays a terminal session:

```
ddd_v1_w_X3r_1432955@runweb64570:~$ cd cs643
ddd_v1_w_X3r_1432955@runweb64570:~/cs643$ ls
key2.pem tanvi_keys_instances2.pem
ddd_v1_w_X3r_1432955@runweb64570:~/cs643$ chmod 400 tanvi_keys_instances2.pem
ddd_v1_w_X3r_1432955@runweb64570:~/cs643$ ssh -i "tanvi_keys_instances2.pem" ec2-user@ec2-54-208-117-172.compute-1.amazonaws.com
The authenticity of host 'ec2-54-208-117-172.compute-1.amazonaws.com (54.208.117.172)' can't be established.
ECDSA key fingerprint is SHA256:ipopGLIsp4fresuJXKhcnP3/UdUj7AaKGOpJ7o.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-54-208-117-172.compute-1.amazonaws.com,54.208.117.172' (ECDSA) to the list of known hosts.
Last login: Tue Oct 18 15:40:37 2022 from ec2-54-201-250-6.us-west-2.compute.amazonaws.com
[ec2-user@ip-172-31-88-177 ~]$
```

Below the terminal is a file browser window showing a single file named "Amazon Linux 2 AMI". At the bottom are "Previous" and "Next" buttons.

## Instance B is running

This screenshot is identical to the one above, showing the AWS Academy Learner Lab interface for Instance B. The terminal session output is the same, indicating that the instance is running and accessible via SSH.

## 10. Create s3 bucket -

The screenshot shows the AWS S3 Management Console. On the left, a sidebar titled "Amazon S3" lists various services like Buckets, Access Points, and Storage Lens. The main area displays a success message: "Successfully created bucket 'njit-cs643-tanvi-bucket'". It also includes a section on optimizing costs and a table of buckets. The table shows one bucket named "njit-cs643-tanvi-bucket" located in "US East (N. Virginia)" with a creation date of "October 19, 2022, 15:04:44 (UTC-04:00)".

## 11. Drag and drop the jars. Upload it then in the s3 bucket.

The screenshot shows the AWS S3 upload interface. It features a large central area for dragging and dropping files, with the placeholder text "Drag and drop files and folders you want to upload here, or choose Add files, or Add folders.". Below this is a table titled "Files and folders (2 Total, 16.3 MB)" showing two files: "ObjectRecognizer.jar" and "TextRecognizer.jar", both 8.1 MB in size. The interface includes sections for "Destination" and "Feedback".

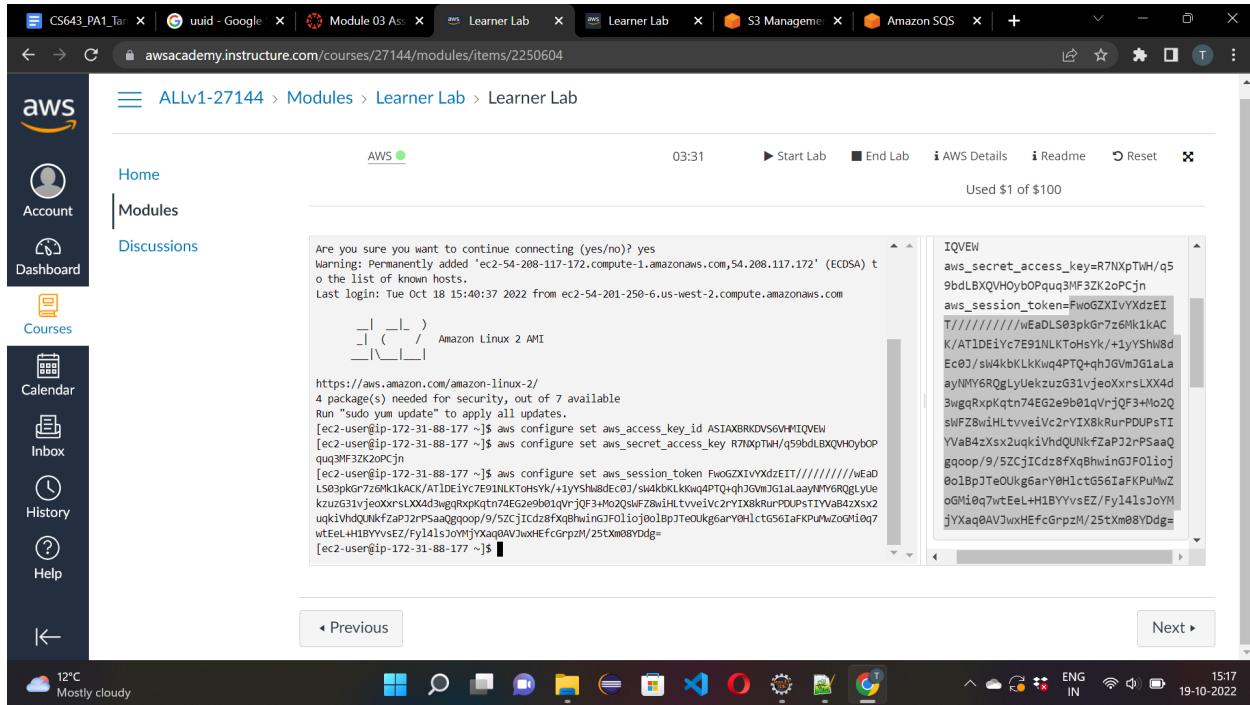
## 12. Install jdk for both instances using this cmd-

```
sudo amazon-linux-extras install java-openjdk11
```

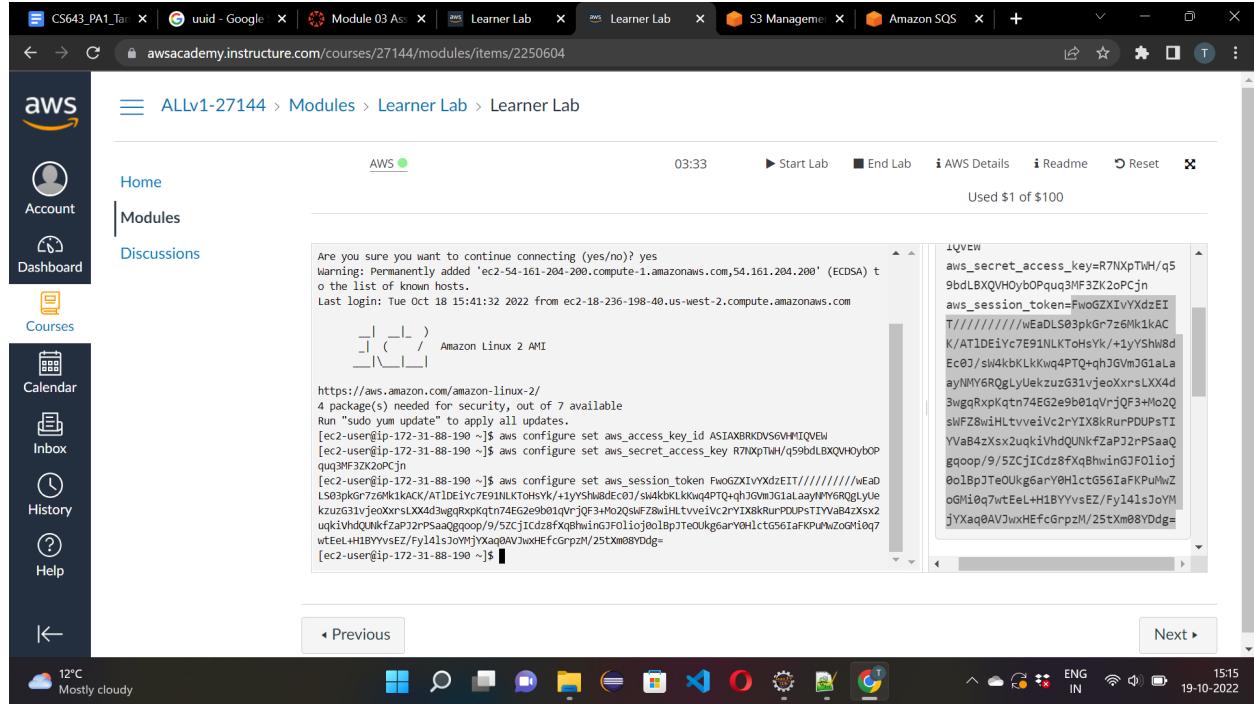
### 13. Set up aws configuration -

```
$ aws configure set aws_access_key_id <<access key >>
$ aws configure set aws_secret_access_key <<secret access key>>
$ aws configure set aws_session_token <<aws session token>>
```

For instance A



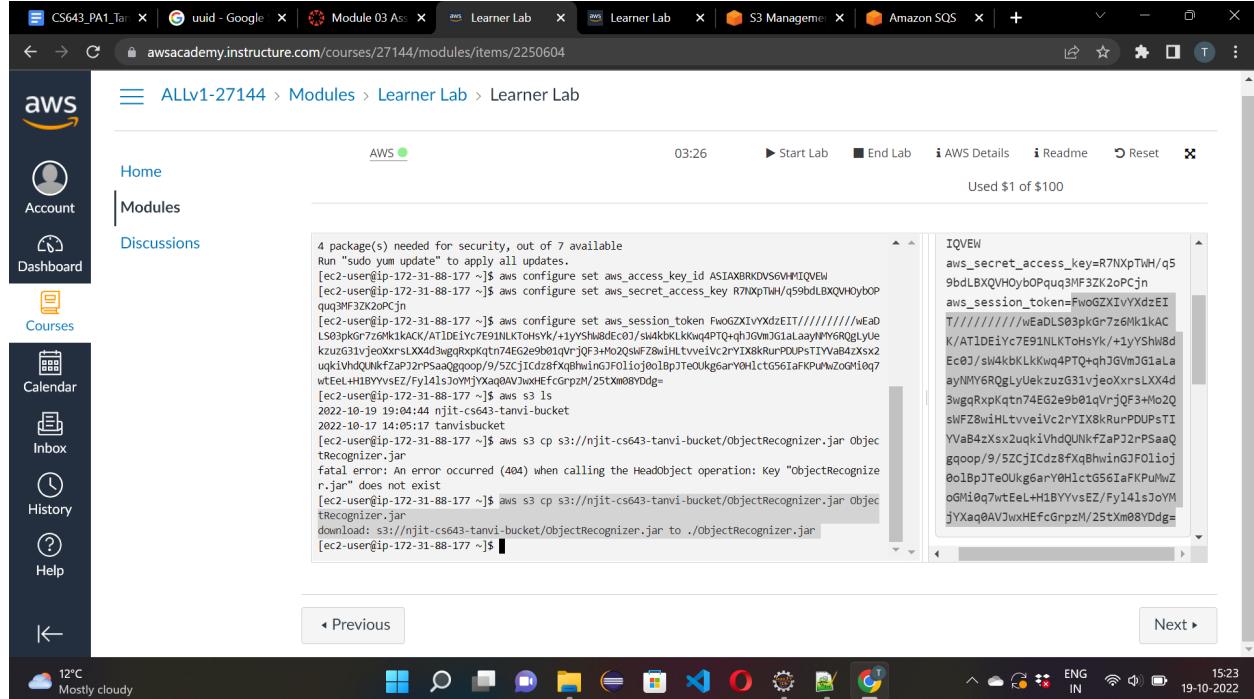
For instance B



14. Copy the required jars to EC2 instance using below command:

## On instance A

```
aws s3 cp s3://njit-cs643-tanvi-bucket/ObjectRecognizer.jar ObjectRecognizer.jar
```



On instance B

```
aws s3 cp s3://njit-cs643-tanvi-bucket/TextRecognizer.jar TextRecognizer.jar
```

Last login: Tue Oct 18 15:41:32 2022 from ec2-18-236-198-40.us-west-2.compute.amazonaws.com

```

[ec2-user@ip-172-31-88-190 ~]$ aws configure set aws_access_key_id ASIAJBRKDVS6VHMIQVEWl
[ec2-user@ip-172-31-88-190 ~]$ aws configure set aws_secret_access_key R7NxpTWh/q59bdLBXQVHOybPquq3MF3ZK2oPCjn
[ec2-user@ip-172-31-88-190 ~]$ aws configure set aws_session_token FwoGZXIVYXdxzEIT//////////wEdLS03pkGr7z6lk1kAC
[ec2-user@ip-172-31-88-190 ~]$ aws configure set aws_region us-east-1
[ec2-user@ip-172-31-88-190 ~]$ aws s3 cp s3://njit-cs643-tanvi-bucket/TextRecognizer.jar TextRec
ognizer.jar
[ec2-user@ip-172-31-88-190 ~]$ java -jar ./TextRecognizer.jar

```

Used \$1 of \$100

15:48 19-10-2022

## 15.Run ObjectRecognizer jar using below command on Instance A:

```
java -jar ./ObjectRecognizer.jar
```

```

[ec2-user@ip-172-31-88-177 ~]$ java -jar ./ObjectRecognizer.jar
Bucket njit-cs-643 exists.
Downloading image no. 1.jpg from bucket njit-cs-643Downloaded image no. 1.jpg
Retrieving image no. 1.jpgfrom S3 bucket
Label for 1.jpg
Car detected on image no. 1.jpg
Downloading image no. 10.jpg from bucket njit-cs-643Downloaded image no. 10.jpg
Retrieving image no. 10.jpgfrom S3 bucket
Label for 10.jpg
Download image no. 2.jpg from bucket njit-cs-643Downloaded image no. 2.jpg
Retrieving image no. 2.jpgfrom S3 bucket
Label for 2.jpg
Car detected on image no. 2.jpg
Download image no. 3.jpg from bucket njit-cs-643Downloaded image no. 3.jpg
Retrieving image no. 3.jpgfrom S3 bucket
Label for 3.jpg
Download image no. 4.jpg from bucket njit-cs-643Downloaded image no. 4.jpg
Retrieving image no. 4.jpgfrom S3 bucket
Label for 4.jpg
Car detected on image no. 4.jpg
Downloading image no. 5.jpg from bucket njit-cs-643Downloaded image no. 5.jpg

```

Used \$1 of \$100

15:48 19-10-2022

```

Car detected on image no. 4.jpg
Downloading image no. 5.jpg from bucket njit-cs-643Downloaded image no. 5.jpg
Retrieving image no. 5.jpgfrom s3 bucket
Label for 5.jpg
Car detected on image no. 5.jpg
Downloading image no. 6.jpg from bucket njit-cs-643Downloaded image no. 6.jpg
Retrieving image no. 6.jpgfrom s3 bucket
Label for 6.jpg
Car detected on image no. 6.jpg
Downloading image no. 7.jpg from bucket njit-cs-643Downloaded image no. 7.jpg
Retrieving image no. 7.jpgfrom s3 bucket
Label for 7.jpg
Car detected on image no. 7.jpg
Downloading image no. 8.jpg from bucket njit-cs-643Downloaded image no. 8.jpg
Retrieving image no. 8.jpgfrom s3 bucket
Label for 8.jpg
Downloading image no. 9.jpg from bucket njit-cs-643Downloaded image no. 9.jpg
Retrieving image no. 9.jpgfrom s3 bucket
Label for 9.jpg
All images have been processed.
[ec2-user@ip-172-31-88-177 ~]$ 

```

Cloud Labs  
Remaining session time: 03:04:14(185 minutes)  
Session started at: 2022-10-11T14:26:07Z  
Session ended at: 2022-10-11T14:26:07Z  
Accumulated lab time: 20:39:00 (1239)

## 16. Run Text Recognizer jar using below command on Instance B:

```
java -jar ./TextRecognizer.jar
```

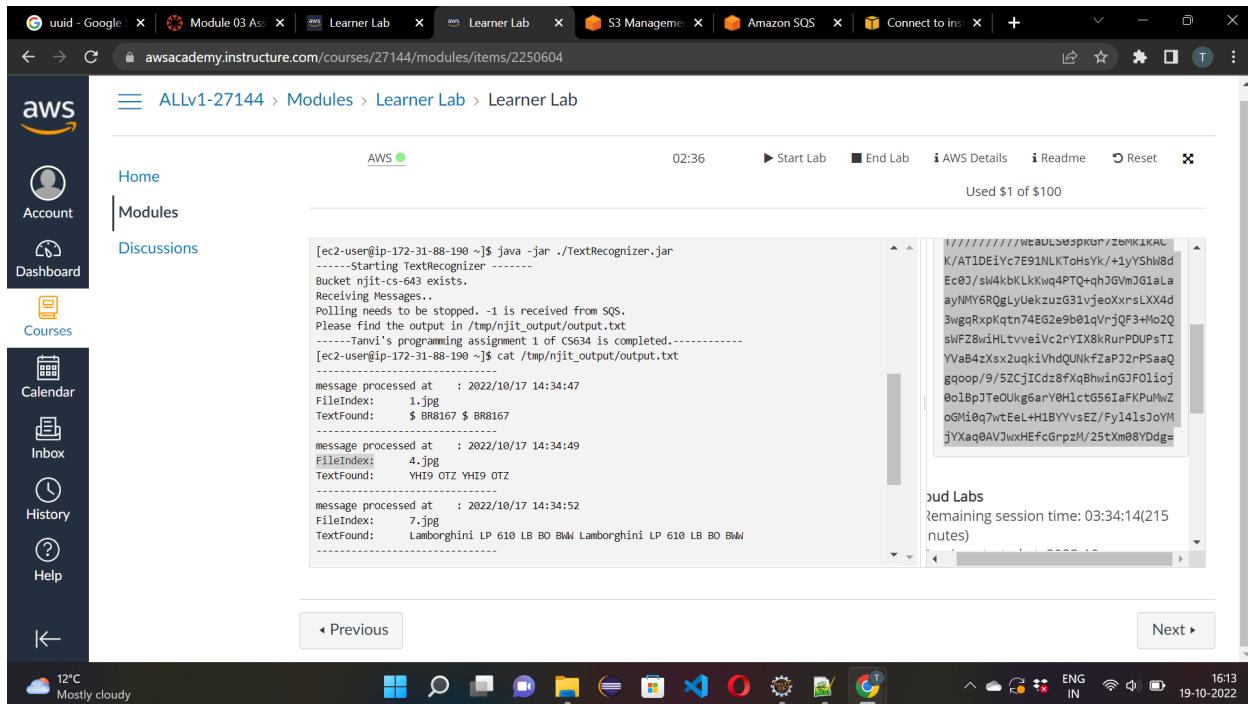
```

4 package(s) needed for security, out of 7 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-88-190 ~]$ aws configure set aws_access_key_id ASIAJBRKDVS6VHMIQVWEw
[ec2-user@ip-172-31-88-190 ~]$ aws configure set aws_secret_access_key R7D0ptWqHq59bJLbXQvHdyb0P
quuMF3ZK2oPcjn
[ec2-user@ip-172-31-88-190 ~]$ aws configure set aws_session_token FwoGZXIvYXdxzEIT//////////wEad
LS83pKgr72gMk1ACKAT1DEiYc7E91NLKToHsyK/+1yYShw8d
kzuG31vje0xrslX4d3wgoLRpxqknt7AE62e9b01qv+j0f3+HcQ2qsfZ8wihLTtveiC2rYX8krurPDUPlsTiYVaB42Xs2
ukqivhdQmkfZapJ2rPsAaQgpoop/95Cj1cdz8fXqPhwinqGfOljoj001bp3TeOUkg6arY0lctG56IafKPUmWzoghi0q7
wtEl+HIBYYvsE/Zfyl4lsJoYXqaqAWJwxEfcGrpz/25txm08YDdg=
[ec2-user@ip-172-31-88-190 ~]$ aws s3 cp s3://njit-cs643-tanvi-bucket/TextRecognizer.jar TextRec
ognizer.jar
download: s3://njit-cs643-tanvi-bucket/TextRecognizer.jar to ./TextRecognizer.jar
[ec2-user@ip-172-31-88-190 ~]$ java -jar ./TextRecognizer.jar
-----Starting TextRecognizer -----
Bucket njit-cs-643 exists.
Receiving Messages...
Polling needs to be stopped. -1 is received from SQS.
Please find the output in /tmp/njit_output/output.txt
-----Tanvi's programming assignment 1 of CS634 is completed.-----
[ec2-user@ip-172-31-88-190 ~]$ 

```

Cloud Labs  
Remaining session time: 03:34:14(215 minutes)

17. Output of TextRecognizer.jar is a text file named output.txt. It will be saved at /tmp/njit-output/.  
 cat /tmp/njit\_output/output.txt

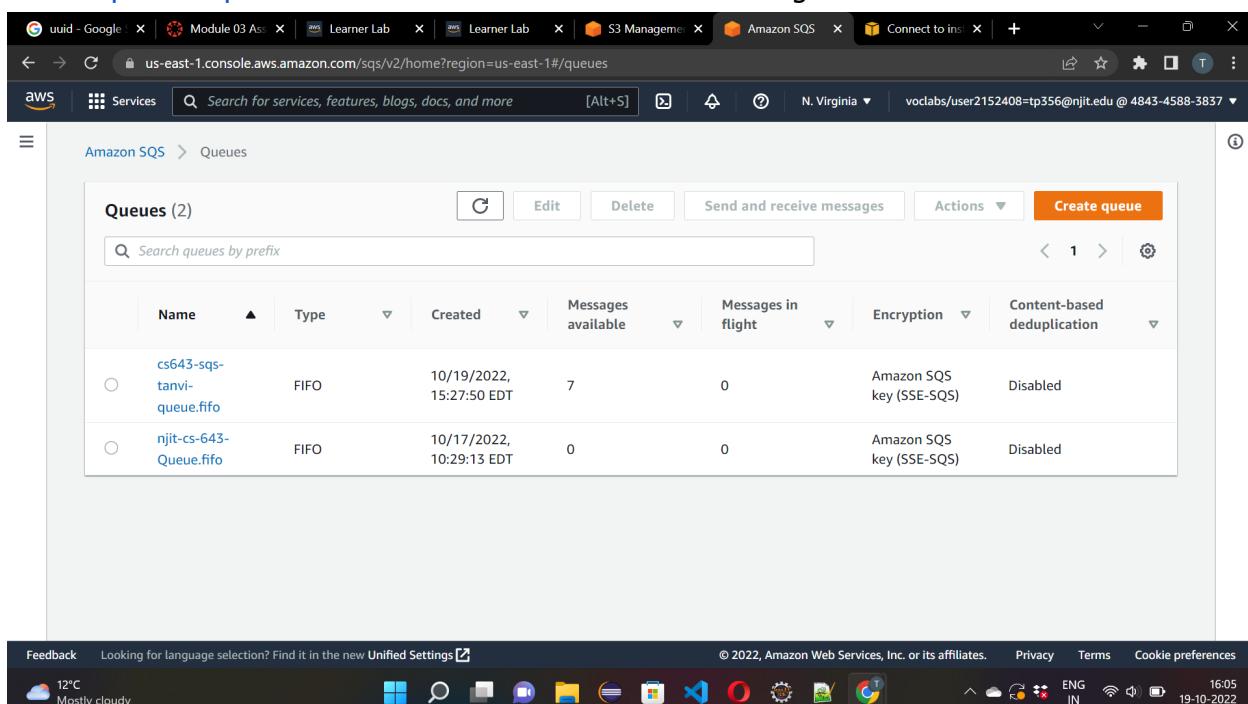


The screenshot shows the AWS Academy Learner Lab interface. On the left, there's a sidebar with various links like Home, Modules, Discussions, Calendar, Inbox, History, Help, and Courses. The main content area displays the terminal output of a Java command:

```
[ec2-user@ip-172-31-88-190 ~]$ java -jar ./TextRecognizer.jar
-----starting TextRecognizer-----
Bucket njit-cs-643 exists.
Receiving Messages...
Polling needs to be stopped. -1 is received from SQS.
Please find the output in /tmp/njit_output/output.txt
-----Tanvi's programming assignment 1 of CS634 is completed.-----
[ec2-user@ip-172-31-88-190 ~]$ cat /tmp/njit_output/output.txt
-----
message processed at : 2022/10/17 14:34:47
FileIndex: 1.jpg
TextFound: $ BR8167 $ BR8167
-----
message processed at : 2022/10/17 14:34:49
FileIndex: 4.jpg
TextFound: YH9 OTZ YH9 OTZ
-----
message processed at : 2022/10/17 14:34:52
FileIndex: 7.jpg
TextFound: Lamborghini LP 610 LB BO BMW Lamborghini LP 610 LB BO BMW
```

The right side of the screen shows a large text area containing a long string of characters, likely the output of the image processing task. Below the terminal output, there's a message about session time.

18. We can see the messages that are passed through the sqs queue. Choose cs643-sqs-tanvi-queue.fifo. Click on Send and Receive messages.



The screenshot shows the AWS SQS service interface. At the top, there's a search bar and a user info bar. The main area is titled "Queues (2)" and contains a table with two rows of data:

Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
cs643-sqs-tanvi-queue.fifo	FIFO	10/19/2022, 15:27:50 EDT	7	0	Amazon SQS key (SSE-SQS)	Disabled
njit-cs-643-Queue.fifo	FIFO	10/17/2022, 10:29:13 EDT	0	0	Amazon SQS key (SSE-SQS)	Disabled

At the bottom, there are navigation links for Feedback, Looking for language selection? Find it in the new Unified Settings, © 2022, Amazon Web Services, Inc. or its affiliates., Privacy, Terms, and Cookie preferences. The status bar also shows weather information and system time.

## Click on send and receive messages

The screenshot shows the AWS SQS console with a FIFO queue named "cs643-sqs-tanvi-queue fifo". The queue details are as follows:

Name	Type	ARN
cs643-sqs-tanvi-queue fifo	FIFO	arn:aws:sqs:us-east-1:484345883837:cs643-sqs-tanvi-queue fifo

Encryption: Amazon SQS key (SSE-SQS)

URL: https://sqs.us-east-1.amazonaws.com/484345883837/cs643-sqs-tanvi-queue fifo

Dead-letter queue: -

Below the details, there are tabs for SNS subscriptions, Lambda triggers, Dead-letter queue, Monitoring, Tagging, Access policy, and Encryption.

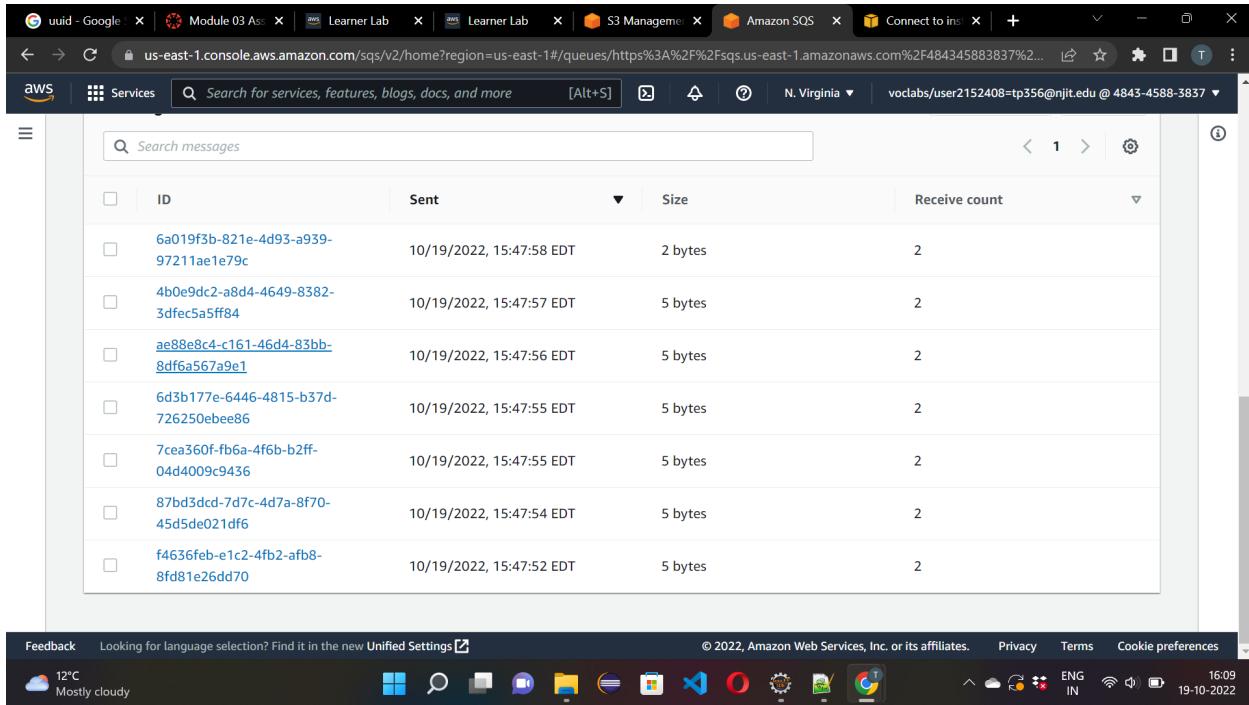
Click on poll for messages. Here we see 7 messages are passed.

The screenshot shows the "Receive messages" section of the AWS SQS console. There are 7 messages available. The settings are as follows:

Messages available	Polling duration	Maximum message count	Polling progress
7	30	10	0 receives/second

Below the settings, there is a table for "Messages (0)" which is currently empty. A "Poll for messages" button is visible at the bottom.

If we click on anyone UUID



The screenshot shows the AWS Lambda console with a successful deployment message:

```
Deployment Succeeded
Function: index
Region: us-east-1
Version: 1
Last Deployment: 2022-10-19T15:47:52Z
```

Below the message, the Lambda function configuration is displayed:

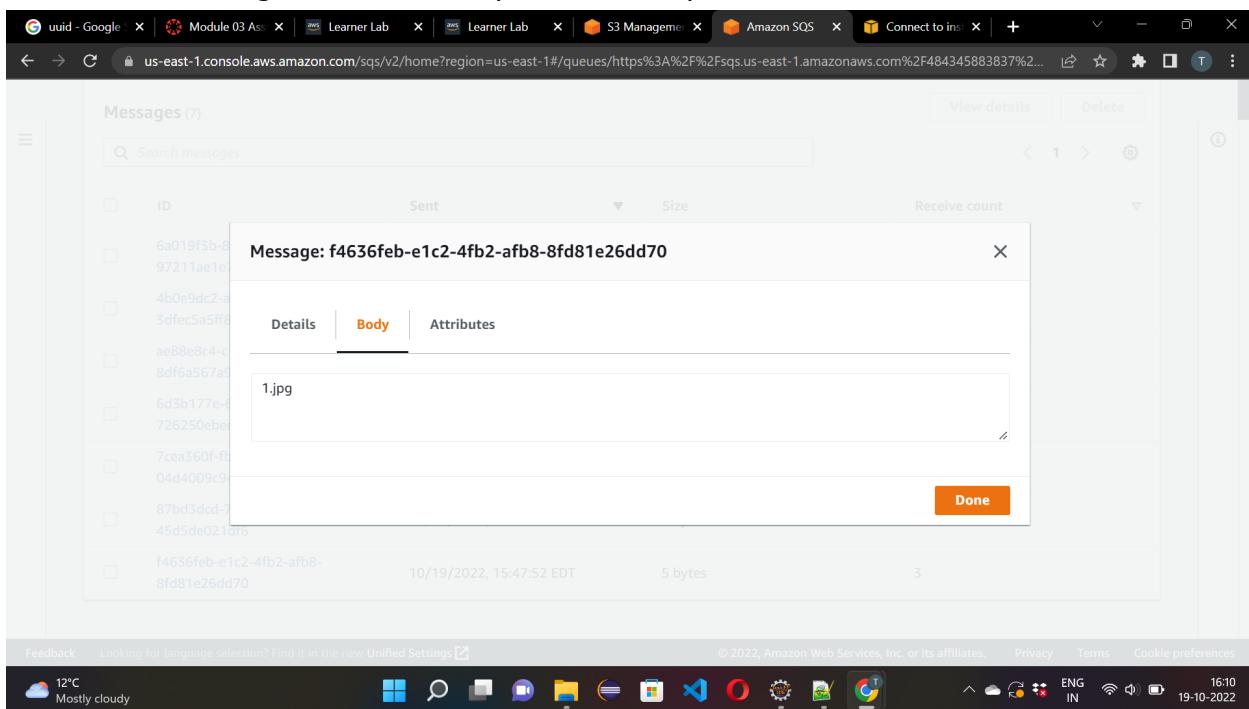
Setting	Value
Code	index (index.js)
Runtime	Node.js 14.x
Memory	128 MB
Timeout	3 seconds
Environment Variables	None
Tracing	None
Logs	Enabled
Deployment	Enabled

The Lambda function has triggered 7 events:

Event Type	Count
CloudWatch Metrics	7

Logs for the function are available in CloudWatch Logs.

We can see the image index that was passed in the queue.



The screenshot shows the AWS SQS console with a list of messages in a queue:

ID	Sent	Size	Receive count
6a019f3b-821e-4d93-a939-97211ae1e79c	10/19/2022, 15:47:58 EDT	2 bytes	2
4b0e9dc2-a8d4-4649-8382-3dfec5a5ff84	10/19/2022, 15:47:57 EDT	5 bytes	2
<u>ae88e8c4-c161-46d4-83bb-8df6a567a9e1</u>	10/19/2022, 15:47:56 EDT	5 bytes	2
6d3b177e-6446-4815-b37d-726250bebe86	10/19/2022, 15:47:55 EDT	5 bytes	2
7cea360f-fb6a-4fb6-b2ff-04d4009c9436	10/19/2022, 15:47:55 EDT	5 bytes	2
87bd3dcd-7d7c-4d7a-8f70-45d5de021df6	10/19/2022, 15:47:54 EDT	5 bytes	2
f4636feb-e1c2-4fb2-afb8-8fd81e26dd70	10/19/2022, 15:47:52 EDT	5 bytes	2

A modal window is open for the message with ID **f4636feb-e1c2-4fb2-afb8-8fd81e26dd70**:

- Details** tab: Shows the message ID and timestamp.
- Body** tab: Displays the file name **1.jpg**.
- Attributes** tab: Shows no attributes.
- Done** button: Closes the modal.

Feedback: Looking for language selection? Find it in the new Unified Settings.

© 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

12°C Mostly cloudy ENG IN 16:09 19-10-2022