AWS-Image-Rekognition:

Built an image recognition pipeline in AWS, using two EC2 instances, S3, SQS, and Rekognition.

Created 2 EC2 instances with Amazon Linux AMI, that will work in parallel. Each instance will run a Java application. Instance A will read 10 images from an S3 bucket and perform object detection in the images. When a car is detected using Rekognition, with confidence higher than 90%, the index of that image is stored in SQS.

Instance B reads indexes of images from SQS as soon as these indexes become available in the queue, and performs text recognition on these images (i.e., downloads them from S3 one by one and uses Rekognition for text recognition).

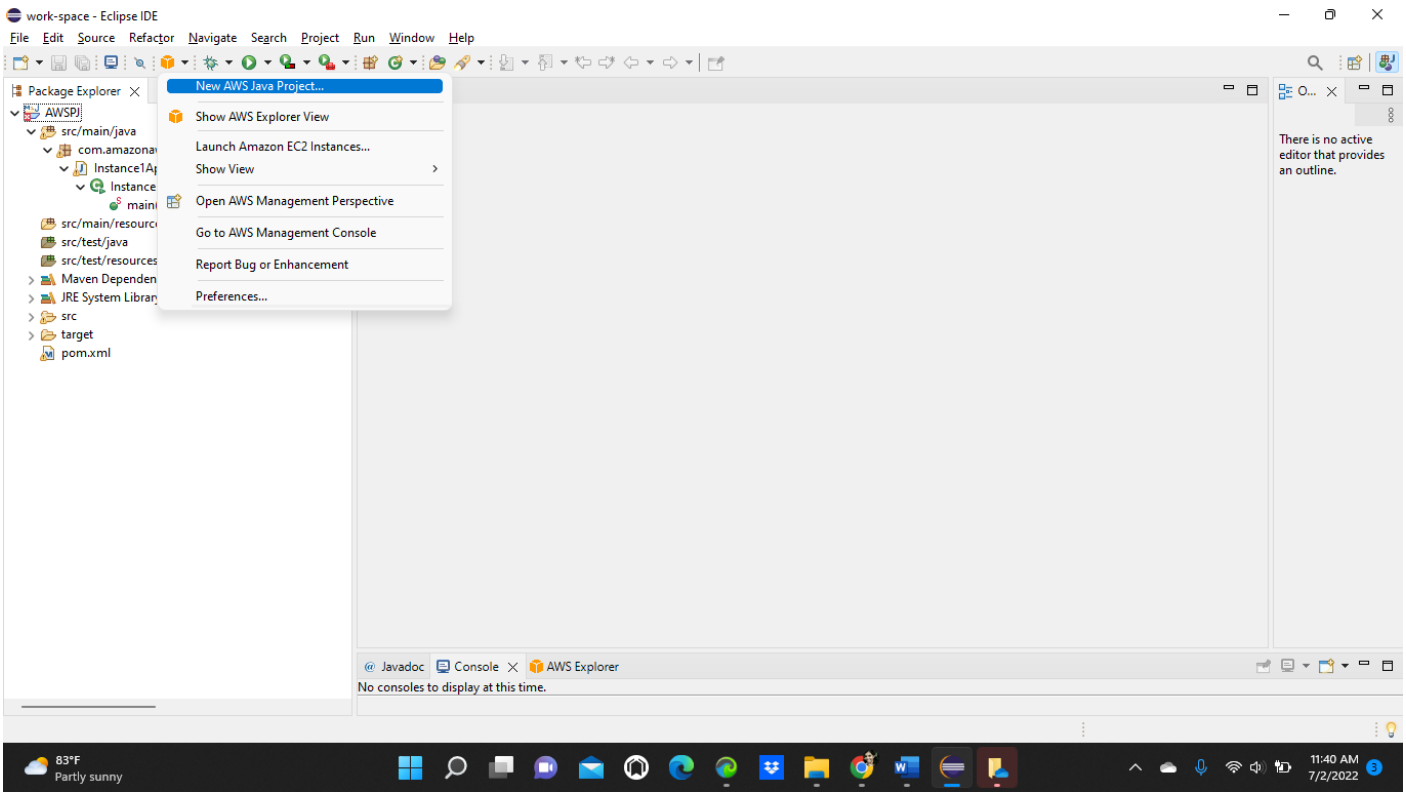
Note that the two instances work in parallel: for example, instance A is processing image 3, while instance B is processing image 1 that was recognized as a car by instance A. When instance A terminates its image processing, it adds index -1 to the queue to signal to instance B that no more indexes will come.

Program output: When instance B finishes, it prints to a file output.txt, in its associated EBS, the indexes of the images that have both cars and text, and also prints the actual text in each image next to its index. Your application must work no matter which instance starts first.

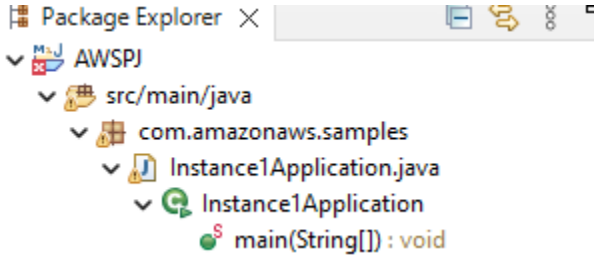
This project is done using Eclipse IDE with java version Java 17 (OpenJDK 17)

You can use any IDE and any java version. Make sure that you use the same version of the java as of your local system in your EC2 instance also.

Part 1: Java code in local system.
Install AWS tool kit from IDE and create a new project under aws toolkit for eclipse.

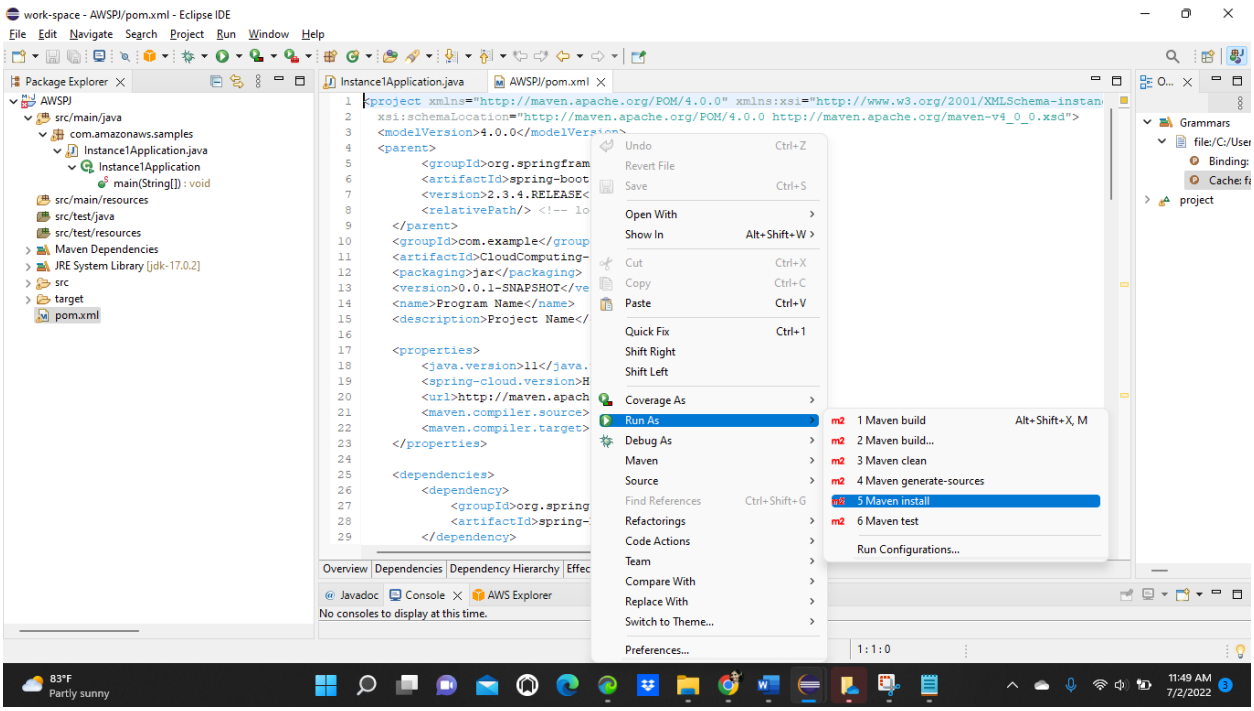


2.Create a new class under com.amazon.samples.



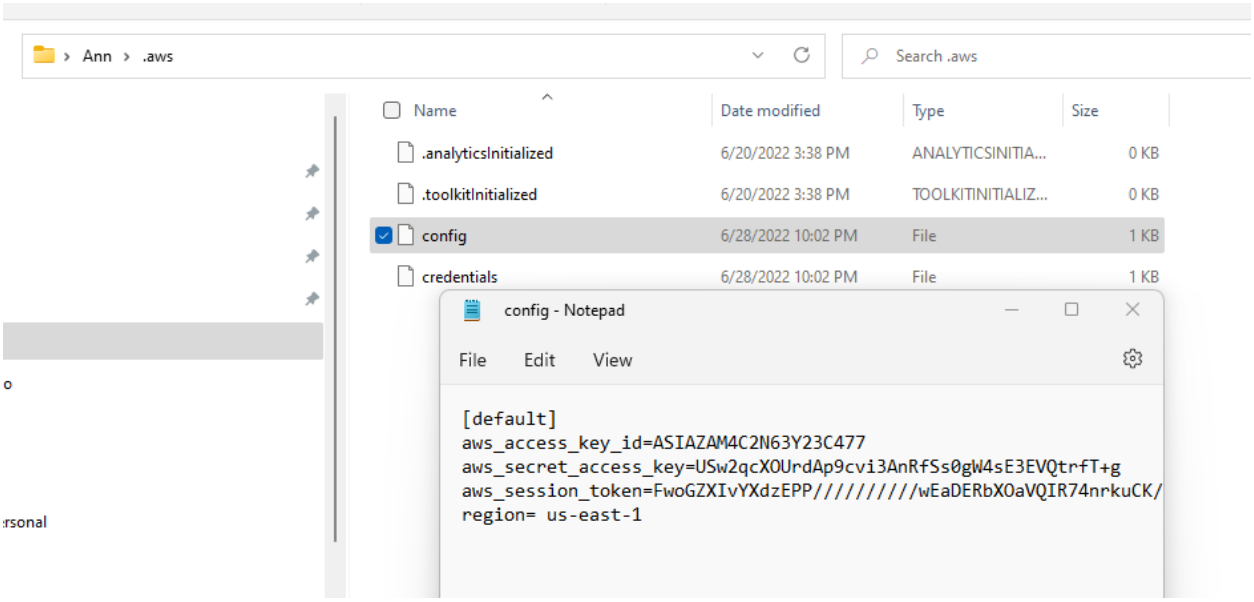
3.Write your code.(uploaded)

4.Upadte your pom file with the required dependencies and do maven install



5. update credentials and configuration file in .aws folder .(C/users/your user/.aws)

aws_access_key_id and aws_secret_access_key can be copied from AWS lab

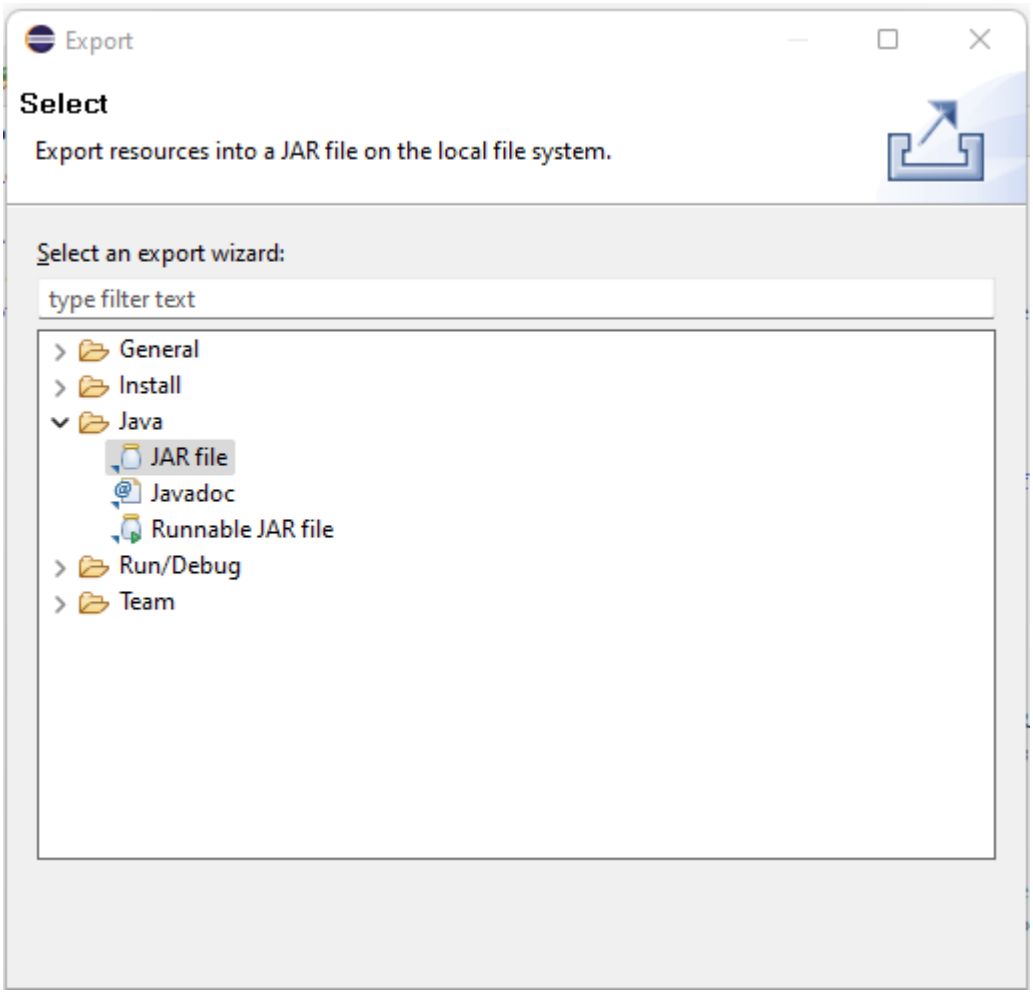


Content of credentials and config are same.

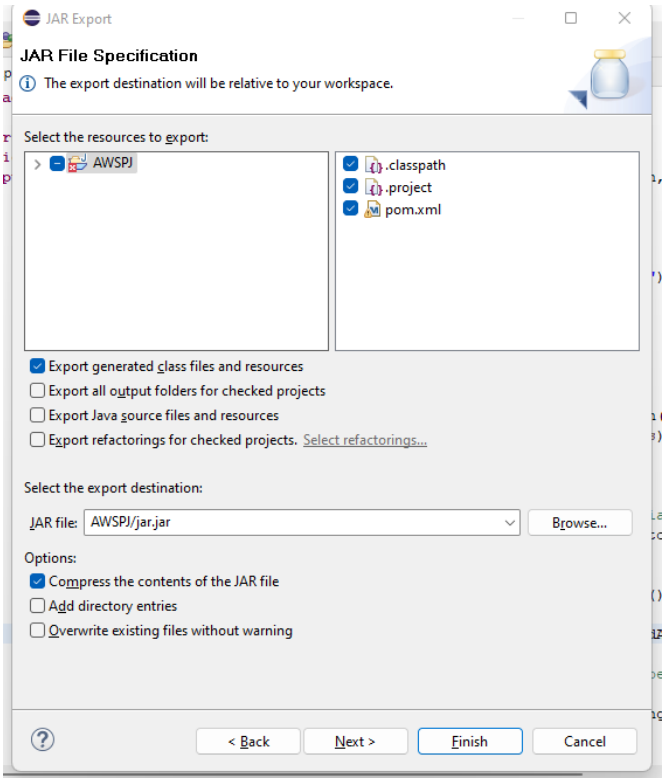
6. Execute and test your code

7. Build the jar file.

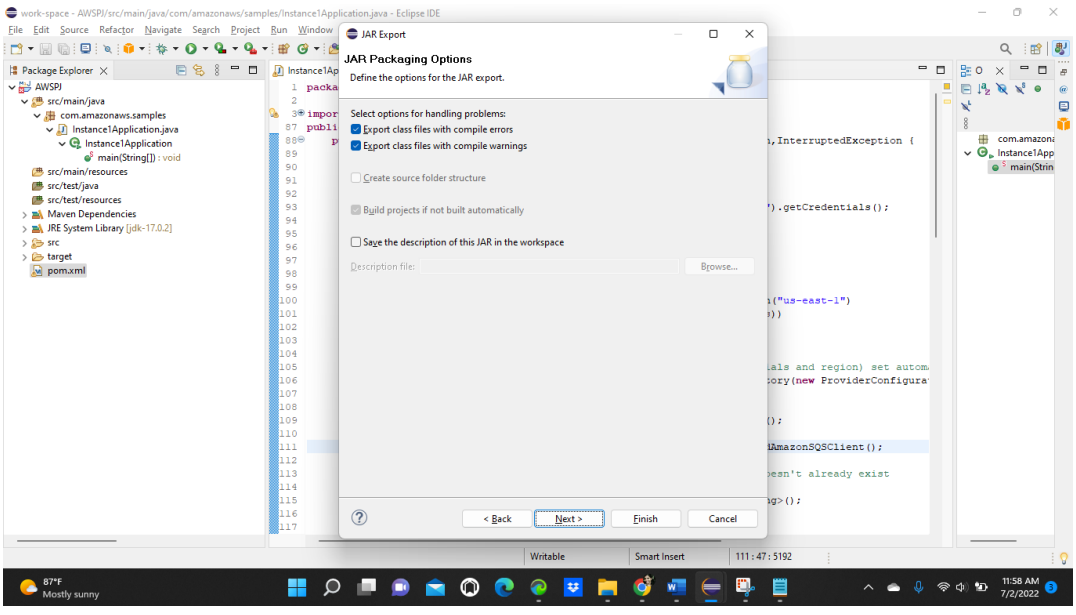
Steps: File→Export→Jar File->Next



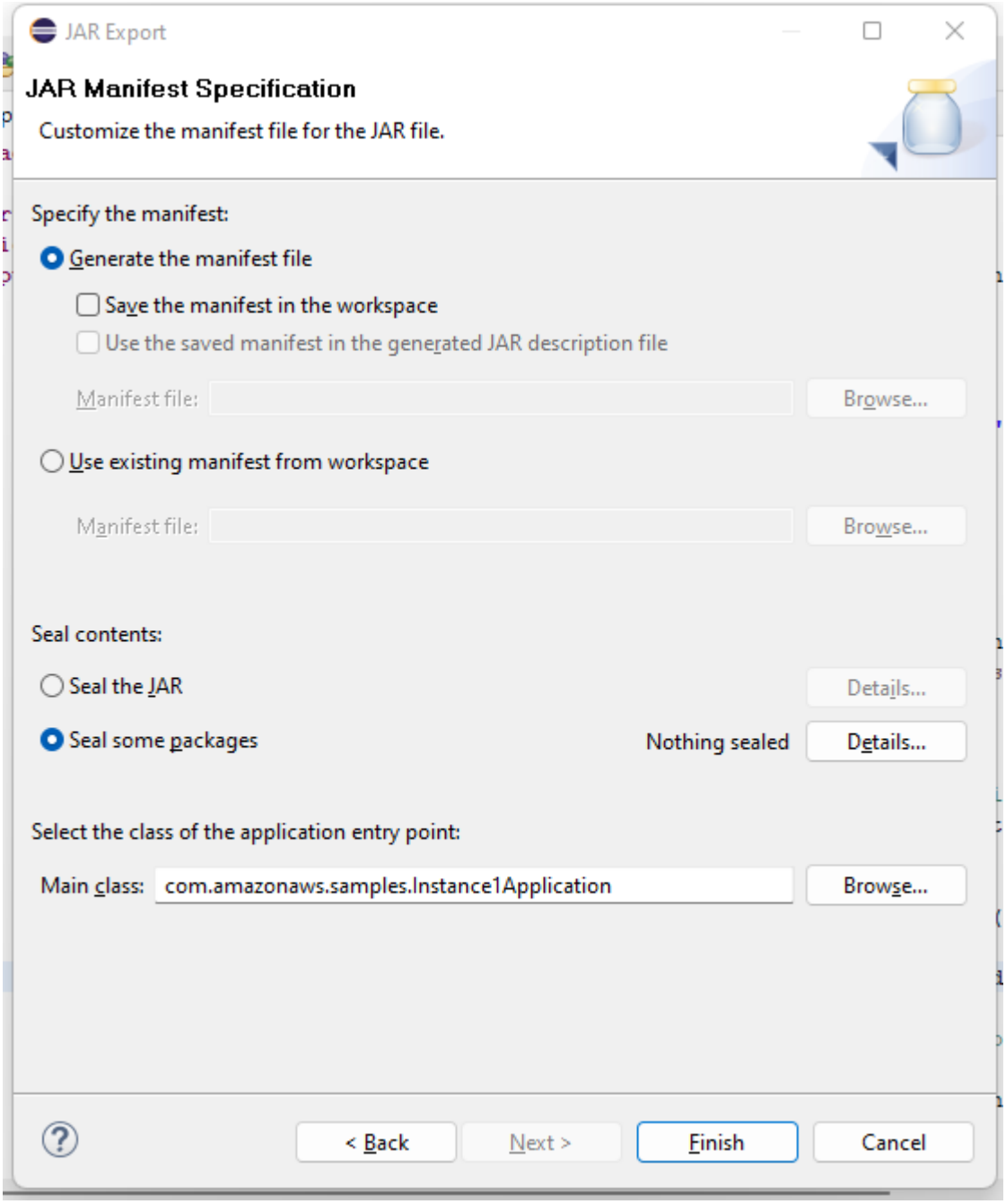
Select your project and select a path→Next



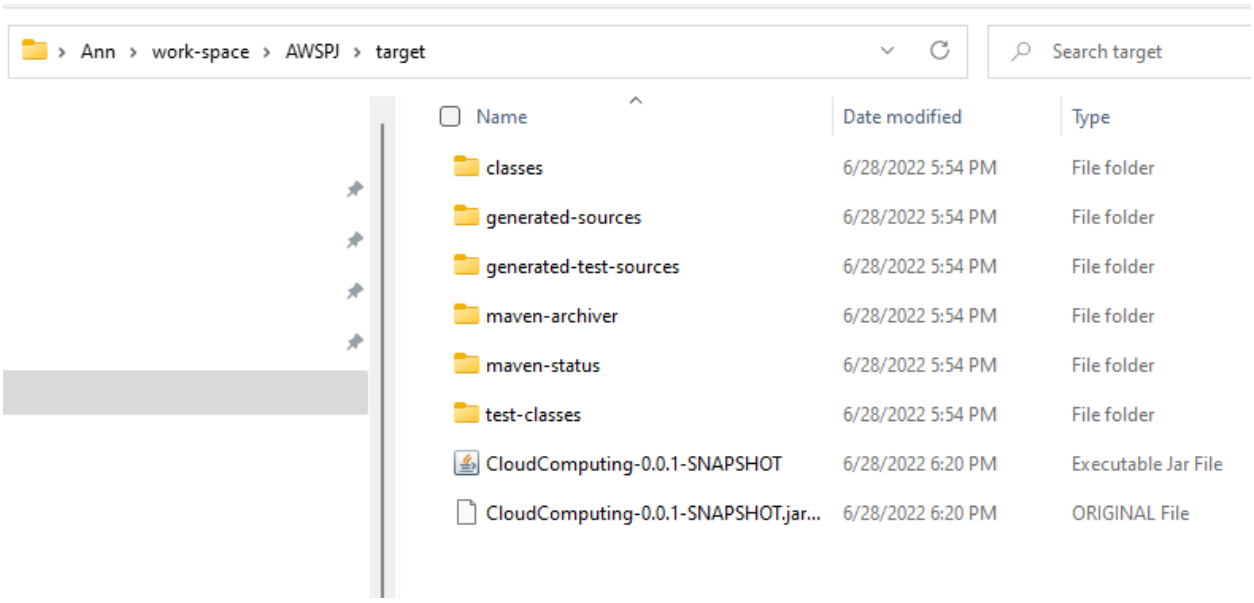
Select Next.



Select your instance and finish.



Make sure that the Manifest file is created under resources folder and chek for jar file under target folder.



8.Create a new class for text recognition code and create jar file.

Part 2. EC2 instance.

1.Create two ec2 instances with aws linux machine and stall both java and maven in ec2 instance. Make sure java version same as that of your java version in local.

Refer below links for steps:

https://techviewleo.com/install-java-openjdk-on-amazon-linux-system/

https://softchief.com/2017/11/07/installing-maven-using-yum-on-ec2-instance-amazon-linux/

2. Upload your instance Ec2

Run below command in command terminal of local system.(change your .pem file path, .jar file path and ec2 instance public key accordingly)

$ scp -i /Users/Ann/file.pem /Users/Ann/cs643-0.0.1-SNAPSHOT.jar ec2-user@ec2-54-167-73-176.compute-1.amazonaws.com:~

3.Connect to ec2 instance.

$ ssh -i "file.pem" ec2-user@ec2-44-202-85-30.compute-1.amazonaws.com

(update the location of .pem file and Public DNS)

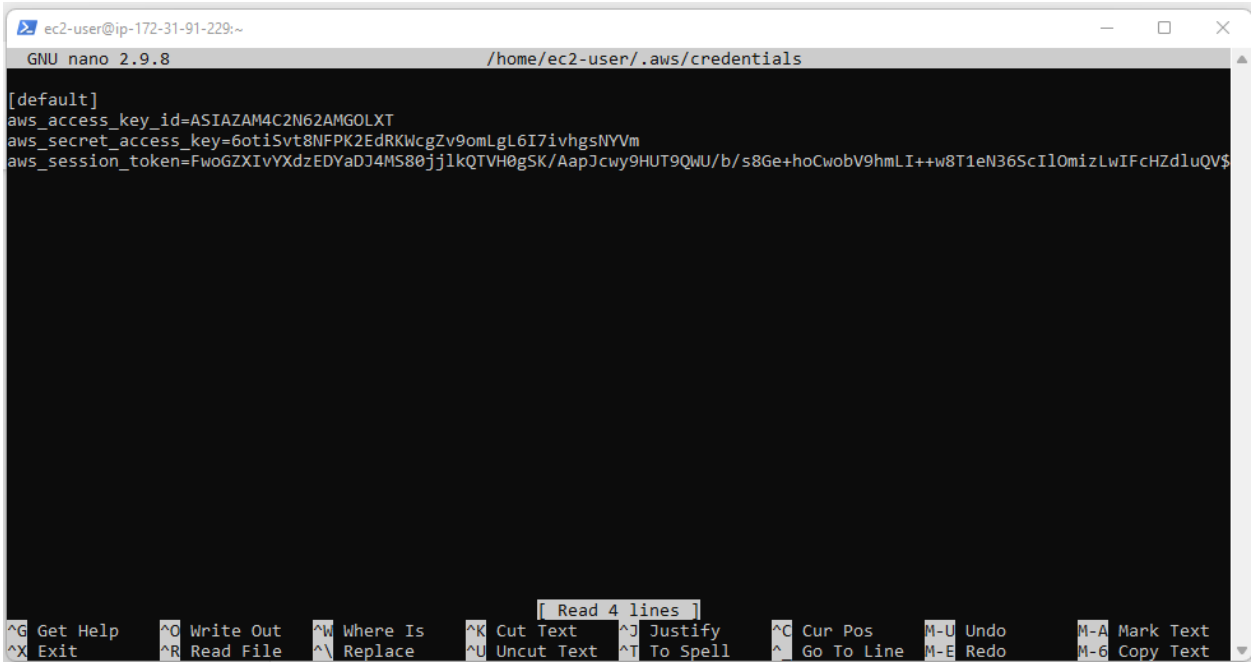
4. Setup config file with vommand $ .aws configure

```
Paste your aws_access_key_id and ‘enter key’

Same was paste the aws_secret_access_key and region. Out file format is text
```

5.set up aws credentials in ec2 instance using below command(this is required only if you are using and educational account. Otherwise you can setup an IAM role).

sudo nano ~/.aws/credentials



copy and paste aws crendiatls including token.

(clt+O, enter, clt+X)

6.Run your code in VM using $ java -jar your jar file name.jar

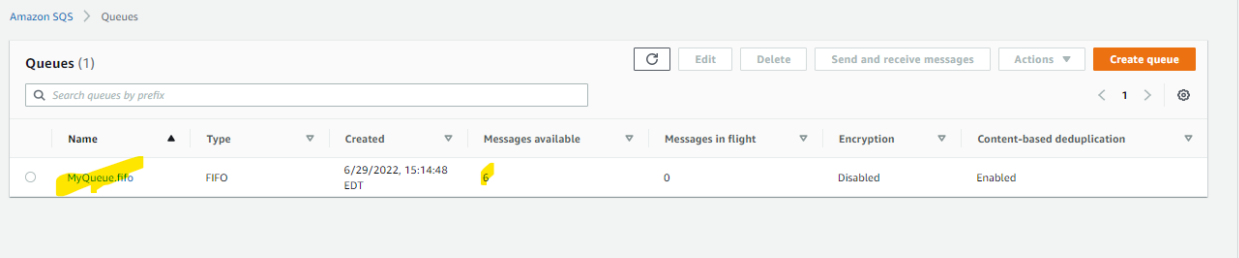
7.output.

```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.3.4.RELEASE)

2022-07-02 16:28:05.358  INFO 7134 --- [           main] c.a.samples.Instance1Application         : Starting Instance1Application v0.0.1-SNAPSHOT on ip-172-31-91-229.ec
2.internal with PID 7134 (/home/ec2-user/CloudComputing-2-0.0.1-SNAPSHOT.jar started by ec2-user in /home/ec2-user)
2022-07-02 16:28:05.365  INFO 7134 --- [           main] c.a.samples.Instance1Application         : No active profile set, falling back to default profiles: default
2022-07-02 16:28:05.721  INFO 7134 --- [           main] c.a.samples.Instance1Application         : Started Instance1Application in 1.446 seconds (JVM running for 3.07)

Listing objects
Detected labels for:  1.jpg => Label: Car ,Confidence: 99.32135
Pushed to sqs.
2022-07-02 16:28:10.375  INFO 7134 --- [           main] c.a.s.javamessaging.SQSMessageProducer   : Message sent to SQS with SQS-assigned messageId: 2fa26441-7bea-4fc0-
9386-c6e7451308ee
JMS Message ID:2fa26441-7bea-4fc0-9386-c6e7451308ee
JMS Message Sequence Number 18870879572043247616
Detected labels for:  2.jpg => Label: Car ,Confidence: 99.96136
Pushed to sqs.
2022-07-02 16:28:11.381  INFO 7134 --- [           main] c.a.s.javamessaging.SQSMessageProducer   : Message sent to SQS with SQS-assigned messageId: 7e615430-5fab-46fc-
b8d1-ea8547969128
JMS Message ID:7e615430-5fab-46fc-b8d1-ea8547969128
JMS Message Sequence Number 18870879572301552384
Detected labels for:  4.jpg => Label: Car ,Confidence: 99.969315
Pushed to sqs.
2022-07-02 16:28:12.418  INFO 7134 --- [           main] c.a.s.javamessaging.SQSMessageProducer   : Message sent to SQS with SQS-assigned messageId: cd46ea4c-6bc1-48c4-
a05a-8d724923110b
JMS Message ID:cd46ea4c-6bc1-48c4-a05a-8d724923110b
JMS Message Sequence Number 18870879572565487616
Detected labels for:  5.jpg => Label: Car ,Confidence: 99.9363
Pushed to sqs.
2022-07-02 16:28:12.974  INFO 7134 --- [           main] c.a.s.javamessaging.SQSMessageProducer   : Message sent to SQS with SQS-assigned messageId: 545037c6-7a51-4e03-
9892-9b720debbd7f
JMS Message ID:545037c6-7a51-4e03-9892-9b720debbd7f
JMS Message Sequence Number 18870879572708591616
Detected labels for:  6.jpg => Label: Car ,Confidence: 99.00655
Pushed to sqs.
2022-07-02 16:28:13.616  INFO 7134 --- [           main] c.a.s.javamessaging.SQSMessageProducer   : Message sent to SQS with SQS-assigned messageId: fe030116-d392-4ddb-
8599-05e450fee369
```

8. Check whether you SQS queue is updated or not(search for sqs in aws services and select Amazon SQS.



9. Same way install java and maven in second instance also. Do all setup and run java code. After execution you can see that its accessing data from queue and SQS queue will be cleared.