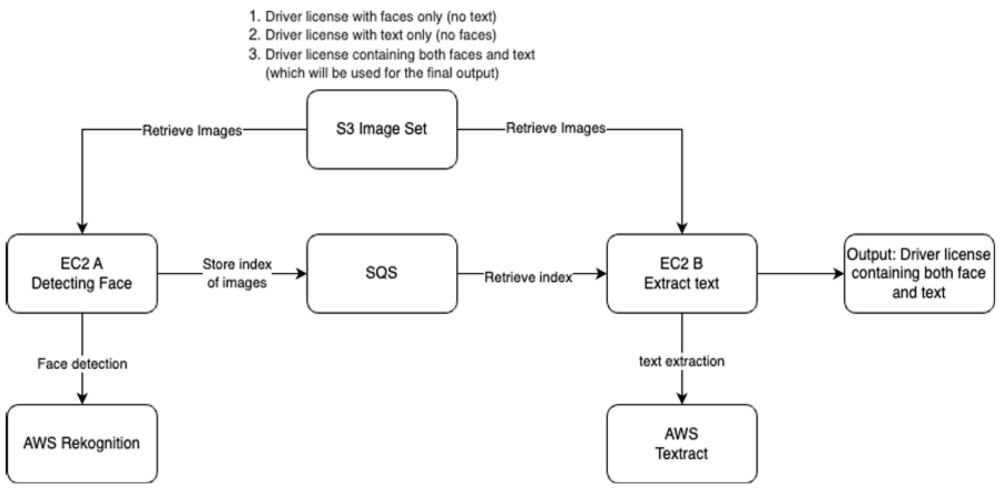
PROGRAMMING ASSIGNMENT 1

By Hemanth Sruthi Vellore

Face and Text Recognition Application

OVERVIEW

This project utilizes AWS services such as Rekognition, S3, SQS, Textract, and EC2 to perform face recognition and text extraction from images. The setup includes two applications, with one for face detection using AWS Rekognition and the other for text extraction using AWS Textract.


1. Driver license with faces only (no text)
2. Driver license with text only (no faces)
3. Driver license containing both faces and text (which will be used for the final output)
S3 Image Set
Retrieve Images
Retrieve Images
EC2 A Detecting Face
Store index of images
SQS
Retrieve index
EC2 B Extract text
Output: Driver license containing both face and text
Face detection
text extraction
AWS Rekognition
AWS Textract

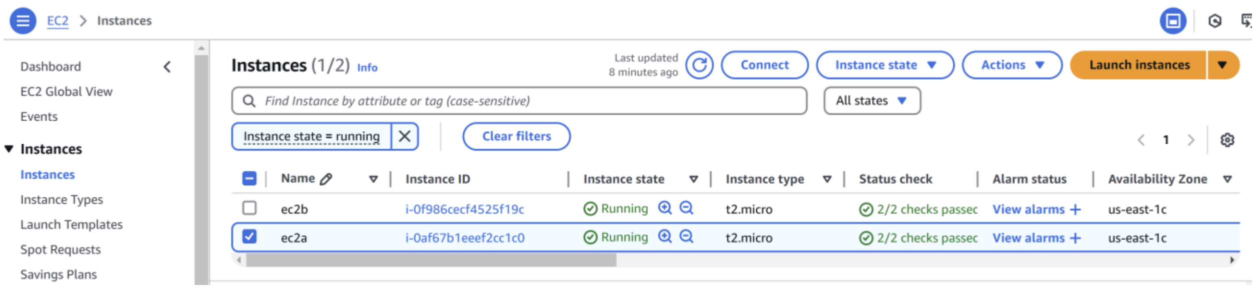
Step 1: Set Up AWS Services

Create EC2 Instances

EC2 Setup [Perform for EC2A and EC2B]

1. **Go to EC2 Dashboard**
   - Click on **"Services"** and then select **"EC2"**.
   - In the **"Instances"** section, click on **"Instances"**.
2. **Launch New Instances**
   - Click on **"Launch Instances"**.
   - Choose the **"Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type."**.
   - Select the **"t2.micro"** instance type.
   - Select the **"vockey "** for Key pair.

- Click "**Next: Configure Security Group**".
    - Click to include the following: **SSH**, **HTTP**, **HTTPS**.
    - For the **Source** drop-down under each rule, select **My IP**.
- Click "**Review and Launch**".


EC2 > Instances

Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans

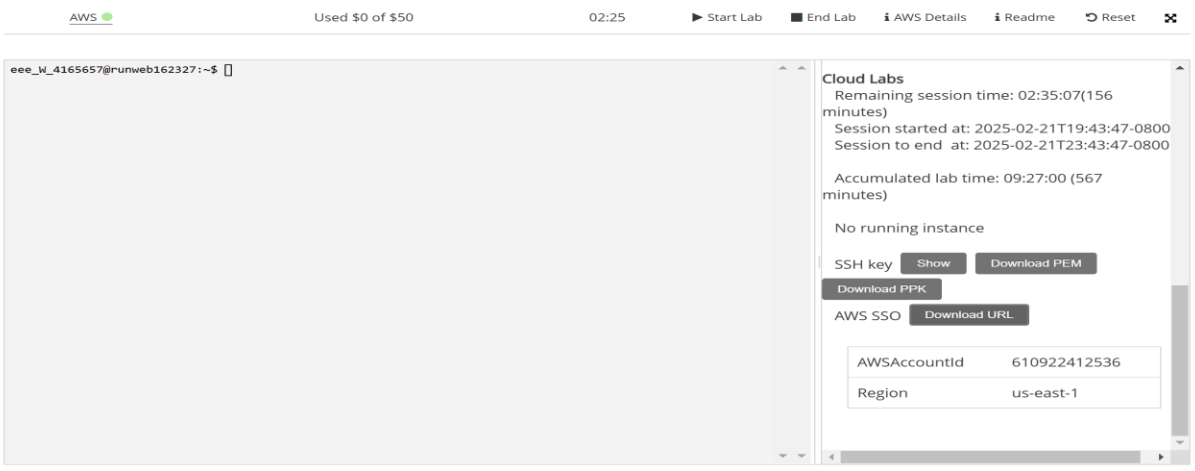
Instances (1/2) Info — Last updated 8 minutes ago — Connect — Instance state — Actions — Launch instances

Find Instance by attribute or tag (case-sensitive) — All states

Instance state = running — Clear filters

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
☐	ec2b	i-0f986cecf4525f19c	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c
☑	ec2a	i-0af67b1eeef2cc1c0	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c

3. **Connect to EC2 Instances Using PuTTY**
    - Open **PuTTY** and enter the **Public IPv4 DNS** of the EC2 instance in the Host Name field (from the EC2 console).


AWS — Used $0 of $50 — 02:25 — Start Lab — End Lab — AWS Details — Readme — Reset

```
eee_W_4165657@runweb162327:~$ 
```

Cloud Labs
Remaining session time: 02:35:07(156 minutes)
Session started at: 2025-02-21T19:43:47-0800
Session to end at: 2025-02-21T23:43:47-0800

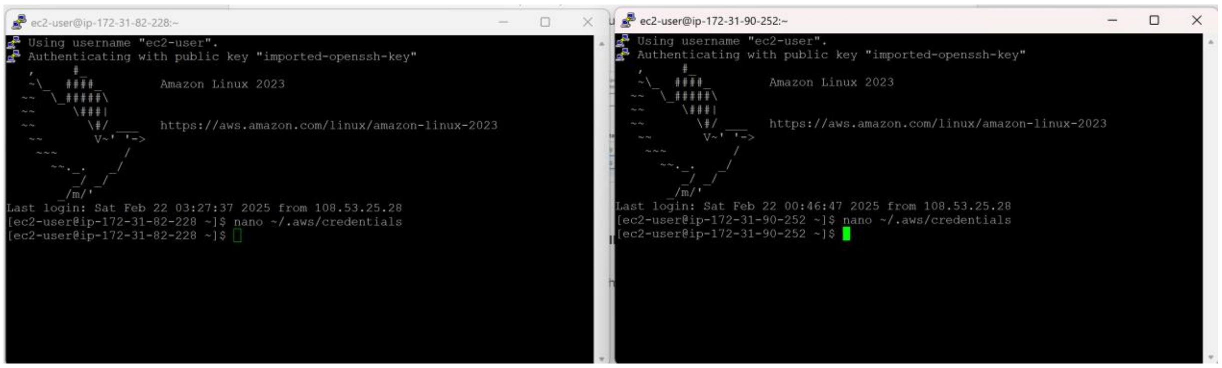
Accumulated lab time: 09:27:00 (567 minutes)

No running instance

SSH key — Show — Download PEM — Download PPK
AWS SSO — Download URL

AWSAccountId	610922412536
Region	us-east-1

- Under the Connection > SSH > Auth section, browse and select the **.ppk file** downloaded from the above screen in **AWS Details** of the **Learner Lab**.
- Click Open to connect.


ec2-user@ip-172-31-82-228:~

```
Using username "ec2-user".
Authenticating with public key "imported-openssh-key"
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Feb 22 03:27:37 2025 from 108.53.25.28
[ec2-user@ip-172-31-82-228 ~]$ nano ~/.aws/credentials
[ec2-user@ip-172-31-82-228 ~]$ 
```

ec2-user@ip-172-31-90-252:~

```
Using username "ec2-user".
Authenticating with public key "imported-openssh-key"
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Feb 22 00:46:47 2025 from 108.53.25.28
[ec2-user@ip-172-31-90-252 ~]$ nano ~/.aws/credentials
[ec2-user@ip-172-31-90-252 ~]$ 
```

4. **Update Java Version on EC2 Instances**

Once logged into each EC2 instance, run the following commands to install Java 1.8:

```
$ sudo yum install java-1.8.0-devel
```

Credentials Setup [Access and Secret Keys]

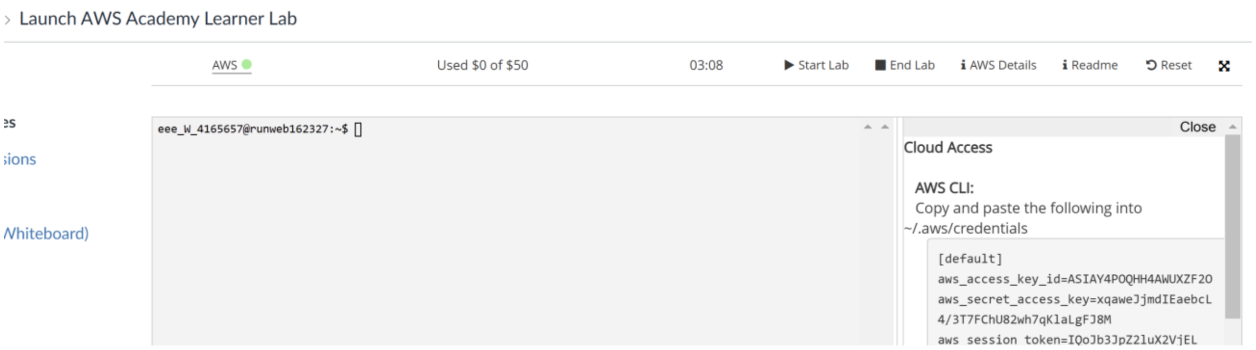
On each EC2 instance, run the following commands to create the AWS credentials file:
bash

```
$ mkdir .aws

$ touch .aws/credentials

$ vi .aws/credentials
```

Paste the following credentials into the `credentials` file (replace with your actual keys):
plaintext

```
[default]

aws_access_key_id=YOUR_ACCESS_KEY_ID

aws_secret_access_key=YOUR_SECRET_ACCESS_KEY
```

Note: You will need to update the credentials on both EC2 instances whenever the keys are rotated (after your session ends).

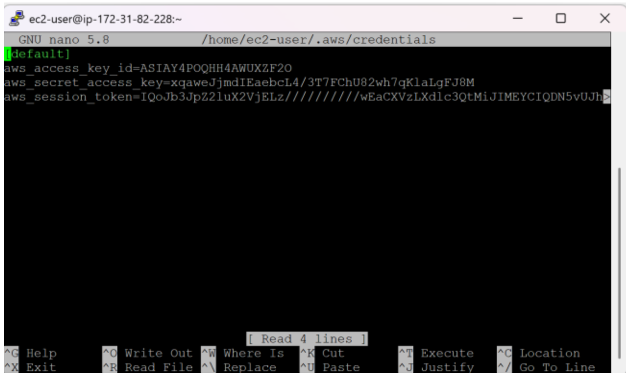


```
eee_W_4165657@runweb162327:~$
```

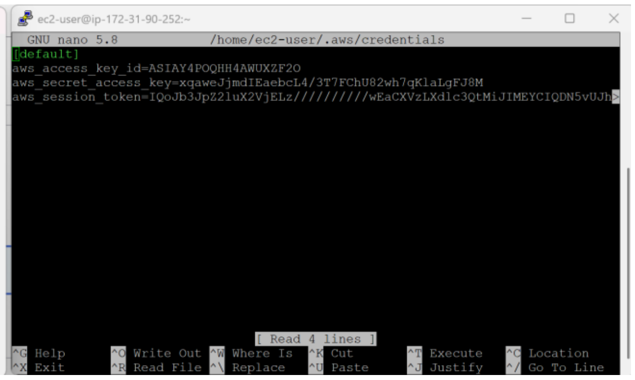
Cloud Access

AWS CLI:
Copy and paste the following into
~/.aws/credentials

```
[default]
aws_access_key_id=ASIAY4POQHH4AWUXZF2O
aws_secret_access_key=xqaweJjmdIEaebcL
4/3T7FChU82wh7qKlaLgFJ8M
aws_session_token=IQoJb3JpZ2luX2VjEL
```



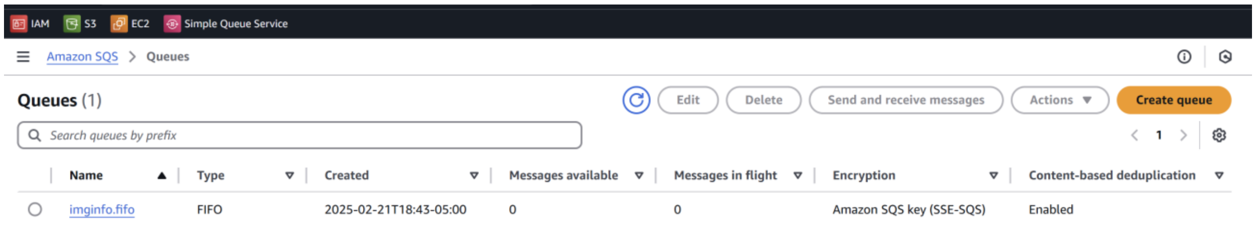
```
  GNU nano 5.8              /home/ec2-user/.aws/credentials
[default]
aws_access_key_id=ASIAY4POQHH4AWUXZF2O
aws_secret_access_key=xqaweJjmdIEaebcL4/3T7FChU82wh7qKlaLgFJ8M
aws_session_token=IQoJb3JpZ2luX2VjELz//////////wEaCXVzLXdlc3QtMiJIMEYCIQDN5vUJh>

                             [ Read 4 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```



```
  GNU nano 5.8              /home/ec2-user/.aws/credentials
[default]
aws_access_key_id=ASIAY4POQHH4AWUXZF2O
aws_secret_access_key=xqaweJjmdIEaebcL4/3T7FChU82wh7qKlaLgFJ8M
aws_session_token=IQoJb3JpZ2luX2VjELz//////////wEaCXVzLXdlc3QtMiJIMEYCIQDN5vUJh>

                             [ Read 4 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

Set Up SQS FIFO Queue

- Go to the **SQS Management Console**.
- Click on **Create Queue**.
- Select **FIFO Queue** and provide a queue name (e.g., imgqueue.fifo).
- Enable Content-based Deduplication.
- Click Create Queue.



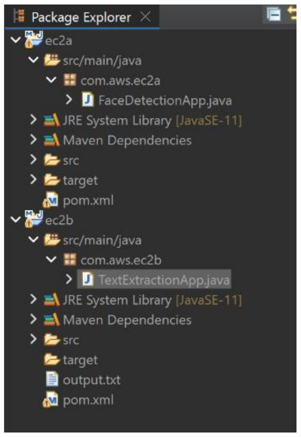
Queues (1)

Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
imginfo.fifo	FIFO	2025-02-21T18:43-05:00	0	0	Amazon SQS key (SSE-SQS)	Enabled

Step 2: Build and Deploy the Maven Java Project

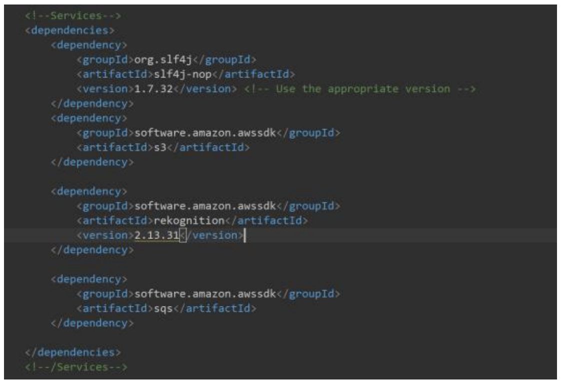
Create Java Project and Convert to JAR in Eclipse

1. **Create a Maven Project in Eclipse**
   - Open **Eclipse** and go to **File** > **New** > **Maven Project**.
   - In the **New Maven Project** dialog, click **Next**.
   - Select **Create a simple project (skip archetype selection)** and click **Next**.
   - Set the **Group Id** (e.g., com.aws) and **Artifact Id** (e.g., ec2a, ec2b), and click **Finish**.



```
ec2a
  src/main/java
    com.aws.ec2a
      FaceDetectionApp.java
  JRE System Library [JavaSE-11]
  Maven Dependencies
  src
  target
  pom.xml
ec2b
  src/main/java
    com.aws.ec2b
      TextExtractionApp.java
  JRE System Library [JavaSE-11]
  Maven Dependencies
  src
  target
  output.txt
  pom.xml
```

2. **Add Dependencies (if needed)**
   - Add external libraries them to the **pom.xml** file.

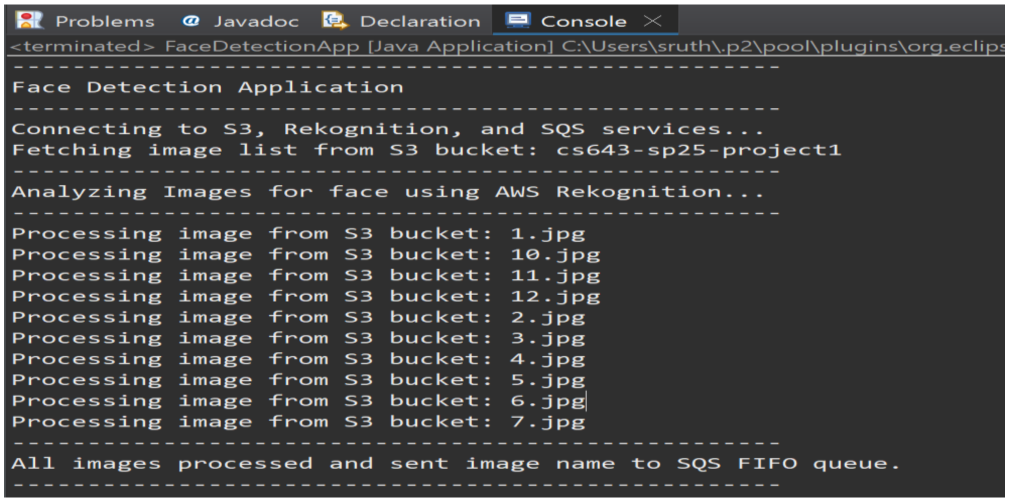


```
<!--Services-->
<dependencies>
    <dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-nop</artifactId>
        <version>1.7.32</version> <!-- Use the appropriate version -->
    </dependency>
    <dependency>
        <groupId>software.amazon.awssdk</groupId>
        <artifactId>s3</artifactId>
    </dependency>

    <dependency>
        <groupId>software.amazon.awssdk</groupId>
        <artifactId>rekognition</artifactId>
        <version>2.13.31</version>
    </dependency>

    <dependency>
        <groupId>software.amazon.awssdk</groupId>
        <artifactId>sqs</artifactId>
    </dependency>

</dependencies>
<!--/Services-->
```

3. **Write Your Java Code**

Navigate to **src/main/java** and create a new Java class (e.g., FaceDetectionApp.java, TextExtractionApp).

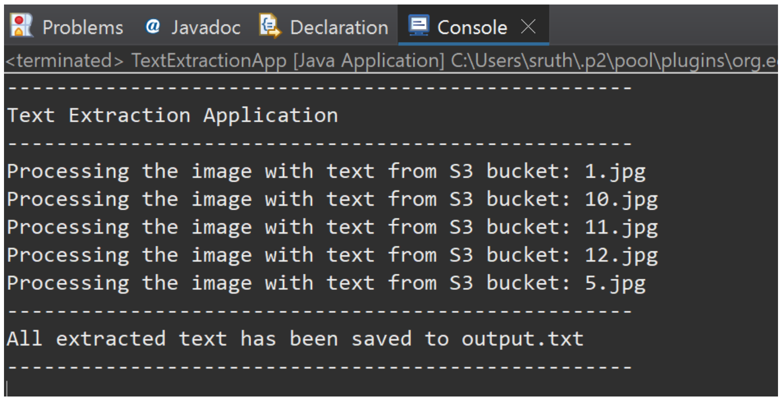
- **EC2 Instance A – Face Detection**
  - Fetches images from an S3 bucket.
  - Uses AWS Rekognition to analyze each image for faces.
  - If a face is detected (75 > confidence), the image index is sent to an SQS FIFO queue.



Problems Javadoc Declaration Console

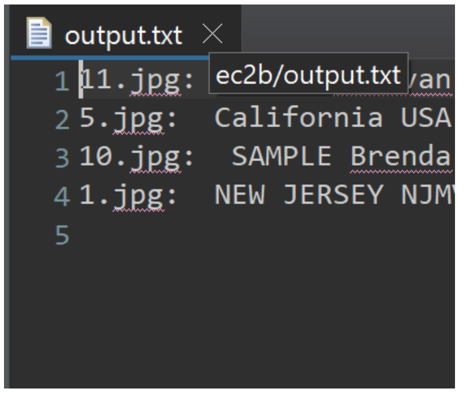
```
<terminated> FaceDetectionApp [Java Application] C:\Users\sruth\.p2\pool\plugins\org.eclips
--------------------------------------------------
Face Detection Application
--------------------------------------------------
Connecting to S3, Rekognition, and SQS services...
Fetching image list from S3 bucket: cs643-sp25-project1
--------------------------------------------------
Analyzing Images for face using AWS Rekognition...
--------------------------------------------------
Processing image from S3 bucket: 1.jpg
Processing image from S3 bucket: 10.jpg
Processing image from S3 bucket: 11.jpg
Processing image from S3 bucket: 12.jpg
Processing image from S3 bucket: 2.jpg
Processing image from S3 bucket: 3.jpg
Processing image from S3 bucket: 4.jpg
Processing image from S3 bucket: 5.jpg
Processing image from S3 bucket: 6.jpg
Processing image from S3 bucket: 7.jpg
--------------------------------------------------
All images processed and sent image name to SQS FIFO queue.
--------------------------------------------------
```

- **EC2 Instance B – Text Extraction**
  - Reads image names from the SQS FIFO queue.
  - Uses AWS Rekognition to extract text from those images.
  - Stores extracted text in a HashMap and writes the results to an output file.
  - Deletes processed messages from the queue.



Problems Javadoc Declaration Console

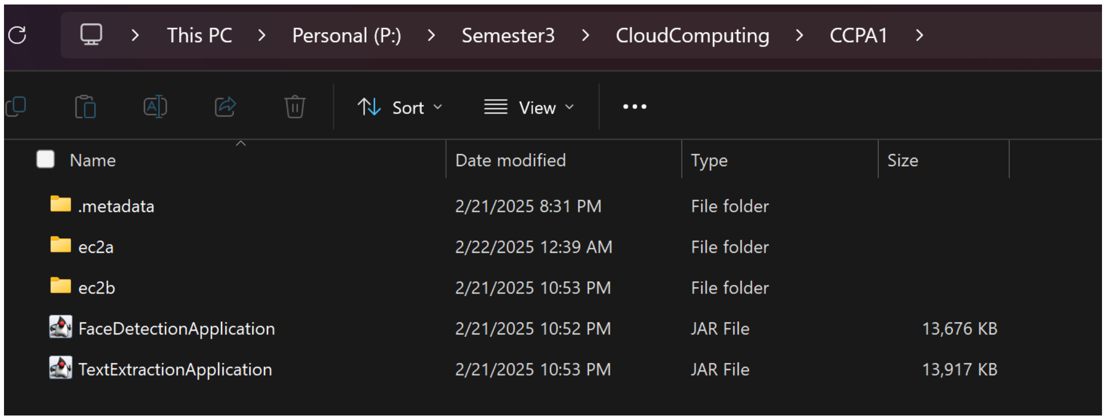
```
<terminated> TextExtractionApp [Java Application] C:\Users\sruth\.p2\pool\plugins\org.e
--------------------------------------------------
Text Extraction Application
--------------------------------------------------
Processing the image with text from S3 bucket: 1.jpg
Processing the image with text from S3 bucket: 10.jpg
Processing the image with text from S3 bucket: 11.jpg
Processing the image with text from S3 bucket: 12.jpg
Processing the image with text from S3 bucket: 5.jpg
--------------------------------------------------
All extracted text has been saved to output.txt
--------------------------------------------------
```



output.txt

```
11.jpg: ec2b/output.txt yan
5.jpg:  California USA
10.jpg:  SAMPLE Brenda
1.jpg:  NEW JERSEY NJM
```

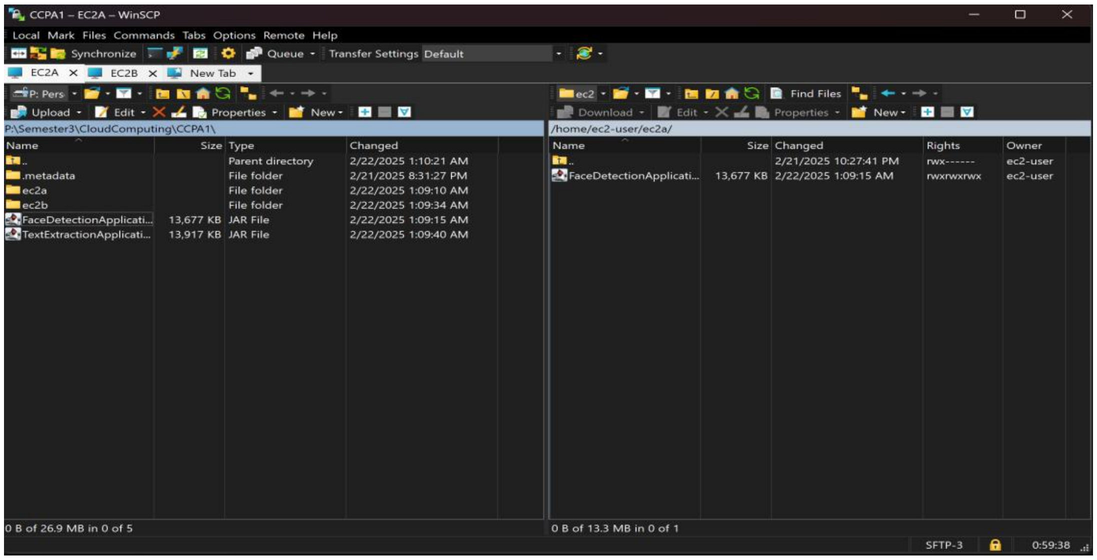
4. **Package Your Application as a JAR File**
   - Right-click on your **project name** in the **Project Explorer**.
   - Select **Run As > Maven build...**.
   - In the **Goals** field, enter clean package and click **Run**.



This PC > Personal (P:) > Semester3 > CloudComputing > CCPA1 >

Name	Date modified	Type	Size
.metadata	2/21/2025 8:31 PM	File folder	
ec2a	2/22/2025 12:39 AM	File folder	
ec2b	2/21/2025 10:53 PM	File folder	
FaceDetectionApplication	2/21/2025 10:52 PM	JAR File	13,676 KB
TextExtractionApplication	2/21/2025 10:53 PM	JAR File	13,917 KB

5. **Deploy JAR File to EC2 Instances**
   - Open **WinSCP** and create a new session:
   - Navigate to the folder on your local machine where the **JAR** file is located.
   - In **WinSCP**, navigate to the directory on the EC2 instance where you want to upload the **JAR** file (typically /home/ec2-user/).
   - Drag and drop the **JAR** file from your local machine to the EC2 instance using **WinSCP**. Run the JAR File on EC2



CCPA1 – EC2A – WinSCP

P:\Semester3\CloudComputing\CCPA1\

Name	Size	Type	Changed
..		Parent directory	2/22/2025 1:10:21 AM
.metadata		File folder	2/21/2025 8:31:27 PM
ec2a		File folder	2/22/2025 1:09:10 AM
ec2b		File folder	2/22/2025 1:09:34 AM
FaceDetectionApplicati...	13,677 KB	JAR File	2/22/2025 1:09:15 AM
TextExtractionApplicati...	13,917 KB	JAR File	2/22/2025 1:09:40 AM

0 B of 26.9 MB in 0 of 5

/home/ec2-user/ec2a/

Name	Size	Changed	Rights	Owner
..		2/21/2025 10:27:41 PM	rwx------	ec2-user
FaceDetectionApplicati...	13,677 KB	2/22/2025 1:09:15 AM	rwxrwxrwx	ec2-user

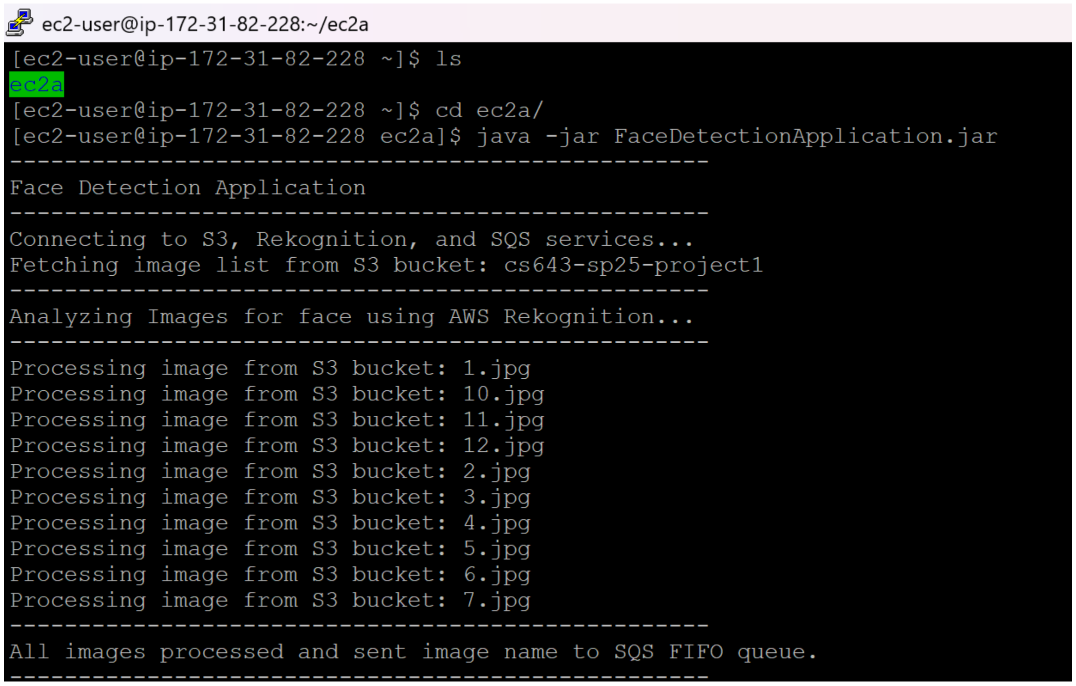
0 B of 13.3 MB in 0 of 1

SFTP-3 0:59:38

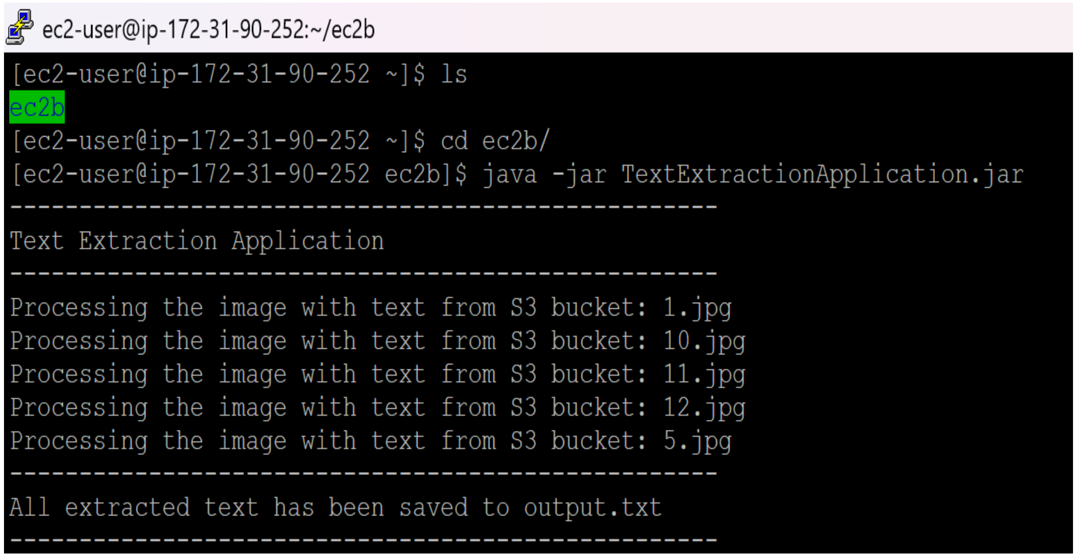
- SSH into each EC2 instance using **PuTTY**.
- Navigate to the directory where you uploaded the JAR file (typically /home/ec2-user/).
- Run the JAR file with the following command:

```
$ java -jar FaceDetectionApp.jar

$ java -jar TextExtractionApp.jar
```

Ensure the **JAR** file is executed successfully on both EC2A and EC2B instances.


ec2-user@ip-172-31-82-228:~/ec2a

```
[ec2-user@ip-172-31-82-228 ~]$ ls
ec2a
[ec2-user@ip-172-31-82-228 ~]$ cd ec2a/
[ec2-user@ip-172-31-82-228 ec2a]$ java -jar FaceDetectionApplication.jar
---------------------------------------------------
Face Detection Application
---------------------------------------------------
Connecting to S3, Rekognition, and SQS services...
Fetching image list from S3 bucket: cs643-sp25-project1
---------------------------------------------------
Analyzing Images for face using AWS Rekognition...
---------------------------------------------------
Processing image from S3 bucket: 1.jpg
Processing image from S3 bucket: 10.jpg
Processing image from S3 bucket: 11.jpg
Processing image from S3 bucket: 12.jpg
Processing image from S3 bucket: 2.jpg
Processing image from S3 bucket: 3.jpg
Processing image from S3 bucket: 4.jpg
Processing image from S3 bucket: 5.jpg
Processing image from S3 bucket: 6.jpg
Processing image from S3 bucket: 7.jpg
---------------------------------------------------
All images processed and sent image name to SQS FIFO queue.
---------------------------------------------------
```

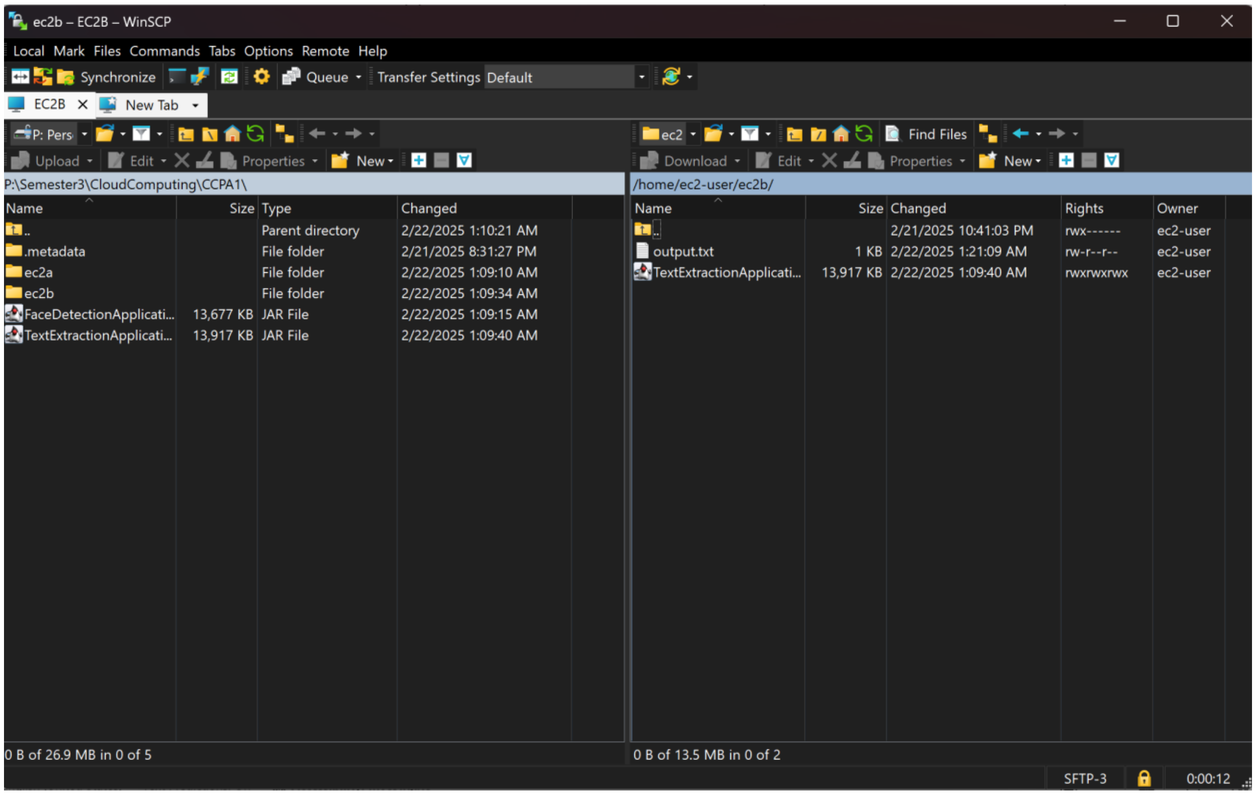

ec2-user@ip-172-31-90-252:~/ec2b

```
[ec2-user@ip-172-31-90-252 ~]$ ls
ec2b
[ec2-user@ip-172-31-90-252 ~]$ cd ec2b/
[ec2-user@ip-172-31-90-252 ec2b]$ java -jar TextExtractionApplication.jar
------------------------------------------------
Text Extraction Application
------------------------------------------------
Processing the image with text from S3 bucket: 1.jpg
Processing the image with text from S3 bucket: 10.jpg
Processing the image with text from S3 bucket: 11.jpg
Processing the image with text from S3 bucket: 12.jpg
Processing the image with text from S3 bucket: 5.jpg
------------------------------------------------
All extracted text has been saved to output.txt
------------------------------------------------
```

6. **Output**

Finally, once both programs have finished running,

A file named output.txt is generated on EC2B in the home directory. This output file will display the indexes of images that contained both face and text, along with the associated text from each image.



P:\Semester3\CloudComputing\CCPA1\

Name	Size	Type	Changed
..		Parent directory	2/22/2025 1:10:21 AM
.metadata		File folder	2/21/2025 8:31:27 PM
ec2a		File folder	2/22/2025 1:09:10 AM
ec2b		File folder	2/22/2025 1:09:34 AM
FaceDetectionApplicati...	13,677 KB	JAR File	2/22/2025 1:09:15 AM
TextExtractionApplicati...	13,917 KB	JAR File	2/22/2025 1:09:40 AM

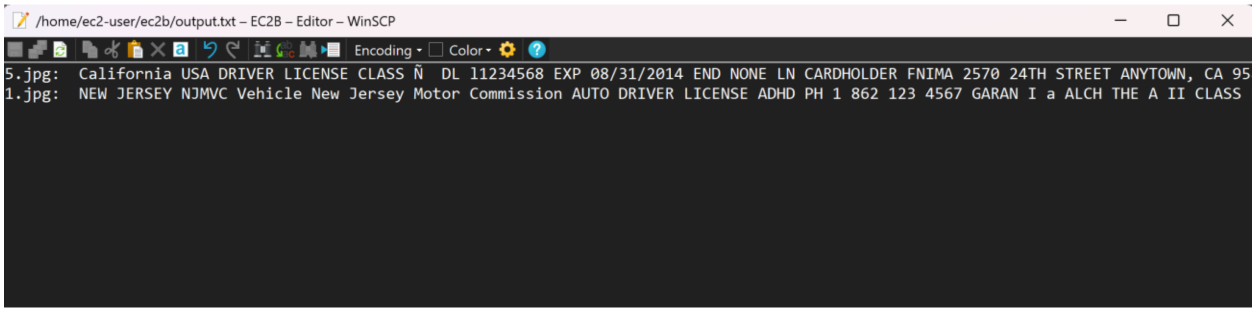
0 B of 26.9 MB in 0 of 5

/home/ec2-user/ec2b/

Name	Size	Changed	Rights	Owner
..		2/21/2025 10:41:03 PM	rwx------	ec2-user
output.txt	1 KB	2/22/2025 1:21:09 AM	rw-r--r--	ec2-user
TextExtractionApplicati...	13,917 KB	2/22/2025 1:09:40 AM	rwxrwxrwx	ec2-user

0 B of 13.5 MB in 0 of 2

SFTP-3 0:00:12



```
5.jpg:  California USA DRIVER LICENSE CLASS Ñ  DL 11234568 EXP 08/31/2014 END NONE LN CARDHOLDER FNIMA 2570 24TH STREET ANYTOWN, CA 95
1.jpg:  NEW JERSEY NJMVC Vehicle New Jersey Motor Commission AUTO DRIVER LICENSE ADHD PH 1 862 123 4567 GARAN I a ALCH THE A II CLASS
```

Github Repository Link

https://github.com/sruthivellore/DriverLicenseDetailsExtraction.git

Video Link

https://youtu.be/ZER3IG9OBrA