

## MANAGED SERVICES ON AWS-PROJECT OPTION 2

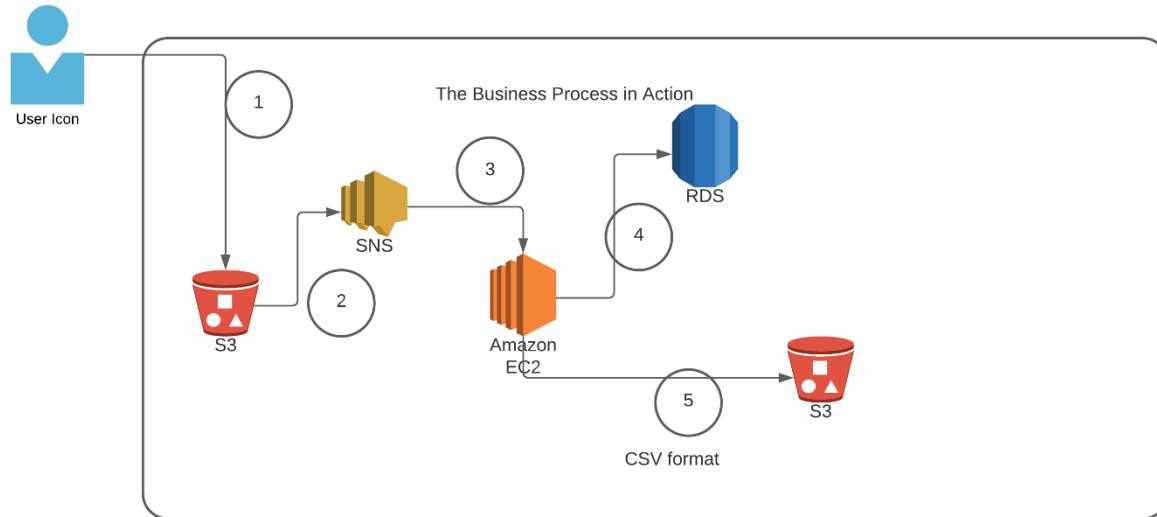
Declaration	
	Questions in this exercise are intentionally complex and could be convoluted or confusing. This is by design and to simulate real life situations where customers seldom give crystal clear requirements and ask unambiguous questions.

I have read the above statement and agree to these conditions	
I AGREE	<b>ADEKUNLE ADEYINKA ADEGBIE</b>
<Enter your name above this line to indicate that you are in agreement>	

Instructions
Every screenshot requested in this workbook is compulsory and carries 1 marks
Your AWS account ID must be clearly visible in every screenshot using the AWS console; missing id or using someone else's id is not permitted. Such cases will be considered as plagiarism and severe penalty will be imposed.
All screenshots must be in the order mentioned under "Expected Screenshots" for every step
DO NOT WAIT UNTIL THE LAST MINUTE. The program office will not extend the project submission deadline under any circumstances.
The file should be renamed in the format BATCH_FIRSTNAME_LASTNAME_PROJECT1. For example: PGPCCMAY18_VIJAY_DWIVEDI_PROJECT1.pdf

Resource Clean Up
Cloud is always pay per use model and all resources/services that we consume are chargeable. Cleaning up when you've completed your lab or project is always necessary. This is true whether you're doing a lab or implementing a project at your workplace.
After completing the lab, make sure to delete each resource created in reverse chronological order. Each AWS Academy session lasts for 4 hours by default, although you can extend a session to run longer by pressing the start button to reset your session timer. At the end of each session, any resources you created in the account will be preserved. Some AWS resources, such as EC2 instances, may be automatically shut down, while other resources, such as RDS instances will be left running.

## Architecture diagram



Architecture Implementation	
1	The customer uploads the invoice data to S3 bucket in a text format as per their guidelines and policies. This bucket will have a policy to auto delete any content that is more than 1 day old (24 hours).
2	An event will trigger in the bucket that will place a message in SNS topic
3	A custom program running in EC2 will subscribe to the SNS topic and get the message placed by S3 event
4	The program will use S3 API to read from the bucket, parse the content of the file and create a CSV record and save the details in an RDS database
5	The program will use S3 API to write CSV record to destination S3 bucket as new S3 object.
Note	The custom program codebase and sample invoice have been shared along with this workbook on the LMS.

## Step 1: SNS and S3 topic creation

Step number	a
Step name	Creation of Source and target buckets
Instructions	<ol style="list-style-type: none"><li>1) Navigate to S3 using the Services button at the top of the screen</li><li>2) Select "Create Bucket"</li><li>3) Enter a source bucket name and use the default options for the rest of the fields</li><li>4) Click on "Create Bucket"</li><li>5) Repeat the above steps to create a target bucket</li></ol>
Expected screenshots	1) Screen showing created S3 source and target buckets

### 1. Creation of Source bucket

The screenshot shows the AWS S3 console interface. At the top, a green banner indicates that a bucket has been successfully created. Below this, the main 'Buckets' page is displayed. On the left, a sidebar menu includes options like 'Access Points', 'Object Lambda Access Points', 'Multi-Region Access Points', 'Batch Operations', 'IAM Access Analyzer for S3', 'Block Public Access settings for this account', 'Storage Lens' (with 'Dashboards' and 'AWS Organizations settings' sub-options), 'Feature spotlight', and 'AWS Marketplace for S3'. The main content area shows an 'Account snapshot' section with a storage lens link and a 'Buckets (4)' table. The table lists four buckets: 'cf-templates-4bkne47lxuo-us-east-1', 'codepipeline-us-east-1-190597825607', 'elasticbeanstalk-us-east-1-380732321839', and 'glt-source-bucket'. The 'glt-source-bucket' row is highlighted, showing its details: US East (N. Virginia) region, creation date of March 23, 2023, and access permissions (Bucket and objects not public). The bottom of the screen shows the Windows taskbar with various pinned icons and system status information.

## 2. Created Source bucket

The screenshot shows the AWS S3 console interface. The left sidebar is titled "Amazon S3" and includes sections for Buckets, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, Block Public Access settings for this account, Storage Lens, Dashboards, AWS Organizations settings, Feature spotlight, and AWS Marketplace for S3. The main content area is titled "glt-source-bucket" and shows the "Objects" tab selected. It displays a message stating "Objects (0)" and "No objects". Below this is a search bar labeled "Find objects by prefix" and a large "Upload" button. At the bottom of the main content area is another "Upload" button. The top navigation bar shows the URL "https://s3.console.aws.amazon.com/s3/buckets/glt-source-bucket?region=us-east-1&tab=objects" and the AWS logo. The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray indicating the date and time.

## 3. Creation of Target bucket

The screenshot shows the AWS S3 console interface. The left sidebar is identical to the previous screenshot. The main content area shows a green success message: "Successfully created bucket "glt-target-bucket"" and "To upload files and folders, or to configure additional bucket settings choose View details." Below this is an "Account snapshot" section with a "View Storage Lens dashboard" button. The "Buckets" section shows a table with 5 entries:

Name	AWS Region	Access	Creation date
cf-templates-4bkne47xiuo-us-east-1	US East (N. Virginia) us-east-1	Bucket and objects not public	February 19, 2023, 21:10:17 (UTC+01:00)
codepipeline-us-east-1-190597825607	US East (N. Virginia) us-east-1	Objects can be public	February 11, 2023, 10:23:57 (UTC+01:00)
elasticbeanstalk-us-east-1-380732321839	US East (N. Virginia) us-east-1	Objects can be public	February 11, 2023, 10:12:25 (UTC+01:00)
glt-source-bucket	US East (N. Virginia) us-east-1	Bucket and objects not public	March 23, 2023, 10:45:37 (UTC+01:00)
glt-target-bucket	US East (N. Virginia) us-east-1	Bucket and objects not public	March 23, 2023, 10:47:48 (UTC+01:00)

The top navigation bar shows the URL "https://s3.console.aws.amazon.com/s3/buckets?region=us-east-1" and the AWS logo. The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray indicating the date and time.

#### 4. Created target bucket

The screenshot shows the AWS S3 console interface. In the top navigation bar, the URL is https://s3.console.aws.amazon.com/s3/buckets/glt-target-bucket?region=us-east-1&tab=objects. The left sidebar is titled 'Amazon S3' and includes sections for Buckets, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, Block Public Access settings for this account, Storage Lens, Dashboards, AWS Organizations settings, Feature spotlight, and AWS Marketplace for S3. The main content area shows the 'glt-target-bucket' page with tabs for Objects, Properties, Permissions, Metrics, Management, and Access Points. The 'Objects' tab is selected, showing a message 'Objects (0)' and a note about granting explicit permissions. Below this is a toolbar with actions like Copy S3 URI, Copy URL, Download, Open, Delete, Actions, and Create folder. A search bar labeled 'Find objects by prefix' is present. A table header for 'Name', 'Type', 'Last modified', 'Size', and 'Storage class' is shown, followed by a message 'No objects' and 'You don't have any objects in this bucket.' At the bottom is a large 'Upload' button. The bottom of the screen shows the Windows taskbar with various pinned icons and system status information.

Step number	B
Step name	Creation of SNS subscription
Instructions	<ol style="list-style-type: none"><li>1) Navigate to SNS -&gt; Topics</li><li>2) Click on "Create Topic"</li><li>3) Enter the following fields Name : S3toEC2Topic The other options can be ignored for now</li><li>4) Click on Create Topic</li></ol>
Expected screenshots	1) Creation of SNS topic

## 5. Creation of SNS Topic

The screenshot shows the AWS SNS console with a green success message at the top: "Topic S3toEC2Topic created successfully. You can create subscriptions and send messages to them from this topic." The main page displays the "Details" section for the topic, including its Name ("S3toEC2Topic"), ARN ("arn:aws:sns:us-east-1:380732321839:S3toEC2Topic"), and Type ("Standard"). Below this, there are tabs for "Subscriptions", "Access policy", "Data protection policy", "Delivery retry policy (HTTP/S)", "Delivery status logging", "Encryption", "Tags", and "Integrations". The "Subscriptions" tab is selected, showing a table with one row: "Create subscription". The table has columns for "ID", "Endpoint", "Status", and "Protocol". A note below the table states, "No subscriptions found. You don't have any subscriptions to this topic." At the bottom of the page, there is a "Create subscription" button. The browser's address bar shows the URL: "https://us-east-1.console.aws.amazon.com/sns/v3/home?region=us-east-1#/topic/arm:aws sns:us-east-1:380732321839:S3toEC2Topic". The operating system taskbar at the bottom includes icons for various applications like File Explorer, Edge, and File History.

Step number	C
Step name	Modification of SNS Access Policy
Instructions	<p>1) Navigate to SNS -&gt; Topics and select the topic created in the previous step</p> <p>2) Note down the ARN shown in the topic details</p> <p>2) Click on Edit and select "Access Policy".</p> <p>3) Replace the text in the JSON editor with the following</p> <pre>{   "Version": "2012-10-17",   "Id": "example-ID",   "Statement": [     {       "Sid": "example-statement-ID",       "Effect": "Allow",       "Principal": {         "AWS": "*"       },       "Action": [         "SNS:Publish"       ],       "Resource": "<b>SNS-topic-ARN</b>",       "Condition": {         "ArnLike": { "aws:SourceArn": "arn:aws:s3:::<b>bucket-name</b>" },         "StringEquals": { "aws:SourceAccount": "<b>bucket-owner-account-id</b>" }       }     }   ] }</pre> <p>4) Replace the bold text with the SNS topic ARN, source bucket name and your AWS account ID respectively.</p> <p>5) Click on Save Changes</p>
Expected screenshots	1) JSON Editor screen

## 6. Modification of SNS Access Policy—“SNS Topic” page

The screenshot shows the AWS SNS console with the topic 'S3toEC2Topic' selected. A success message at the top indicates 'Topic S3toEC2Topic saved successfully.' The 'Details' section shows the topic's name, ARN, and type. Below this are tabs for 'Subscriptions', 'Access policy', 'Data protection policy', 'Delivery retry policy (HTTP/S)', 'Delivery status logging', 'Encryption', 'Tags', and 'Integrations'. The 'Subscriptions' tab is active, showing a table with one row: 'No subscriptions found'. A button labeled 'Create subscription' is visible. The bottom of the screen shows the Windows taskbar with various pinned icons.

## 7. Modification of SNS Access Policy—“JSON Editor screen 1”

The screenshot shows the AWS SNS JSON editor screen. At the top, there is a note about encryption: 'Amazon SNS provides in-transit encryption by default. Enabling server-side encryption adds at-rest encryption to your topic.' Below this, the 'Access policy - optional' section is expanded, showing the following JSON policy:

```
1 [ "Version": "2012-10-17", "Id": "example-ID", "Statement": [ { "Sid": "example-statement-ID", "Effect": "Allow", "Principal": { "AWS": "*" }, "Action": "SNS:Publish", "Resource": "arn:aws:sns:us-east-1:380732321839:S3toEC2Topic", "Condition": { "StringEquals": { "aws:SourceAccount": "380732321839" } } } ] }
```

Below the policy, there are sections for 'Data protection policy - optional', 'Delivery retry policy (HTTP/S) - optional', and 'Delivery status logging - optional'. The bottom of the screen shows the Windows taskbar with various pinned icons.

## 8. Modification of SNS Access Policy - "JSON Editor screen 2"

The screenshot shows the AWS SNS console with the JSON editor open for modifying the access policy of a topic. The policy is defined as follows:

```

9     "AWS": "*",
10    },
11    "Action": "SNS:Publish",
12    "Resource": "arn:aws:sns:us-east-1:38073231839:S3toEC2Topic",
13    "Condition": {
14      "StringEquals": {
15        "aws:SourceAccount": "38073231839"
16      },
17      "ArnLike": {
18        "aws:SourceArn": "arn:aws:s3:::glt-source-bucket"
19      }
20    }
21  }
22 ]
23

```

The interface includes sections for Encryption (optional), Access policy (optional), Data protection policy (optional), Delivery retry policy (HTTP/S) (optional), and Delivery status logging (optional). The bottom of the screen shows the Windows taskbar with various pinned icons.

Step number	D
Step name	Configuring SNS notifications for S3
Instructions	<p>1) Navigate to S3 and select the source bucket created in Step 1 (a)</p> <p>2) Select Properties and scroll down to Event Notifications and select it</p> <p>3) Select "Create Event Notification"</p> <p>4) Fillup the details as follows</p> <p>Name : S3PutEvent</p> <p>Select PUT from the list of radio buttons</p> <p>Destination : Select SNS Topic</p> <p>SNS : Select S3ToEC2Topic</p> <p>5) Save Changes</p>
Expected screenshots	1) Event Configuration Screen

## 9. Configuration of SNS notification for S3-“creation of event notification”

The screenshot shows the AWS S3 console with the URL <https://s3.console.aws.amazon.com/s3/buckets/glt-source-bucket?region=us-east-1&tab=properties>. A green success message at the top states "Successfully created event notification '\$S3PutEvent'. Operation successfully completed." The page displays the properties of the bucket "glt-source-bucket". Under the "Bucket overview" section, it shows the AWS Region as "US East (N. Virginia) us-east-1", the Amazon Resource Name (ARN) as "arn:aws:s3:::glt-source-bucket", and the Creation date as "March 23, 2023, 10:45:37 (UTC+01:00)". The "Properties" tab is selected. In the "Bucket Versioning" section, there is a note about versioning and an "Edit" button. Below that, it shows "Bucket Versioning" status as "Disabled". There is also a note about Multi-factor authentication (MFA) delete. The bottom of the screen shows the Windows taskbar with various pinned icons and system status information.

Step 2: Run the custom program in the EC2 instance

Step number A

Step name Creation of the EC2 instance and RDS instance

Instructions  
1) Navigate to EC2 -> Instances  
2) Create an EC2 instance with the following parameters

AMI : Amazon Linux 2

VPC : Default

Security group : Ports 22 and 8080 should be opened

3) Navigate to RDS

4) Create an RDS instance with the following parameters:

Engine type : MySql

Template : Dev/Test

Set the username and password as required

DB Instance class : Burstable

Instance type : t3.micro

Public Access : Yes

## VPC Security group : Create New ()

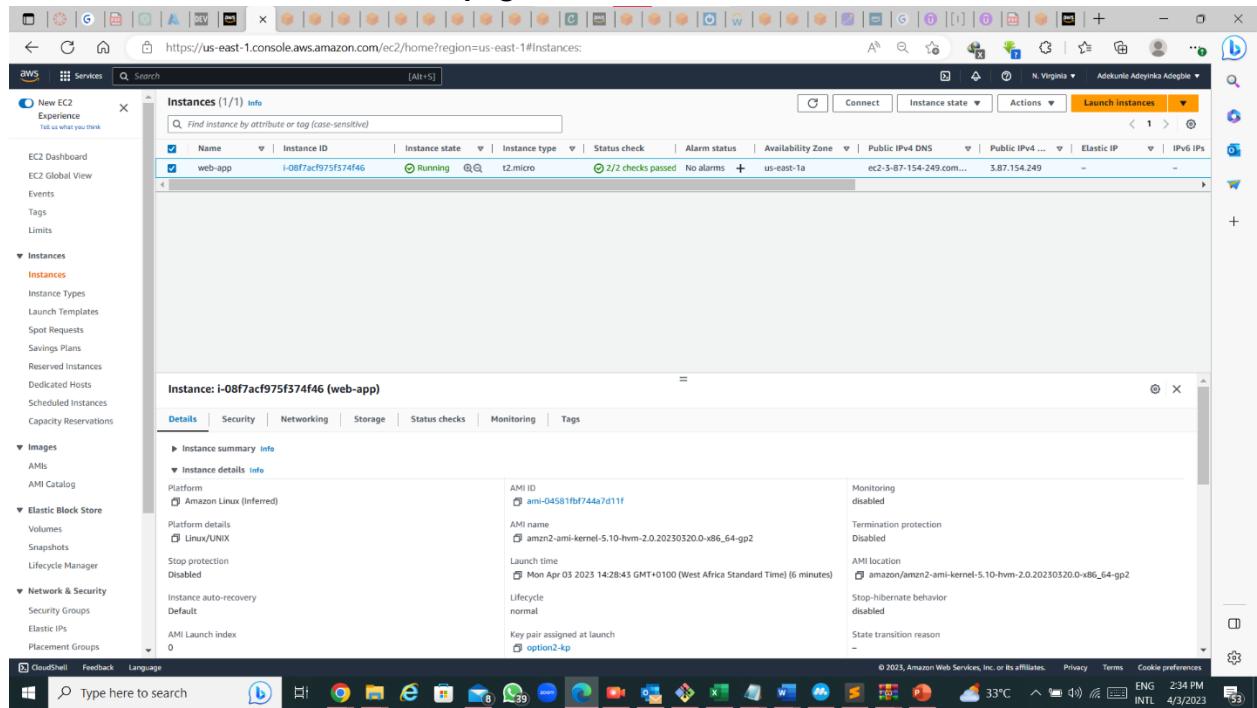
Under Additional Configuration, add an initial database name. Take note of this name as it will be required later.

Uncheck “Enable Enhanced Monitoring”

Ensure that the security group created by the RDS deployment has port 3306 open for all incoming connections from all sources.

- |                      |   |
|----------------------|---|
| Expected screenshots | 1) List of instances after creation of EC2 instance<br>2) List of RDS instances |
|----------------------|---|

## 10. Launched EC2 Instance-“web-page”



## 11. Launched RDS Instance-“glt-db-instance”

The screenshot shows the AWS RDS console with the URL <https://us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#databases>. The left sidebar is titled 'Amazon RDS' and includes sections for Dashboard, Databases, Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Events, Event subscriptions, Recommendations, and Certificate update. The main content area is titled 'Creating database glt-db-instance' and displays a message: 'Your database might take a few minutes to launch.' Below this is a note: 'How was your experience creating an Amazon RDS database? Provide feedback' and a link to the 'Aurora User Guide'. The 'Databases' section shows a table with one row for 'glt-db-instance'. The table columns include DB identifier, Role, Engine, Region & AZ, Size, Status, Actions, CPU, Current activity, Maintenance, and VPC. The 'Status' column shows 'Creating'. The 'Actions' button is highlighted in orange. The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray.

## 12. RDS Instance page-“glt-db-instance”

The screenshot shows the AWS RDS instance details page for 'glt-db-instance' with the URL <https://us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#database:id=glt-db-instance;is-cluster=false>. The left sidebar is identical to the previous screenshot. The main content area is titled 'glt-db-instance' and includes a 'Summary' section with details: DB identifier (glt-db-instance), CPU (-), Status (Creating), Class (db.t3.micro), Role (Instance), Current activity (-), Engine (MySQL Community), and Region & AZ (-). Below the summary are tabs for Connectivity & security, Monitoring, Logs & events, Configuration, Maintenance & backups, and Tags. The 'Connectivity & security' tab is selected and shows the Endpoint, Networking, and Security sections. The Endpoint section lists 'Endpoint' and 'Port'. The Networking section lists 'Availability Zone' (-) and 'VPC' (vpc-0dc027bc3c9cfab5b4). The Subnet group section lists 'default-vpc-0dc027bc3c9cfab5b4'. The Subnets section lists 'subnet-0fb751ec59ff5d2d', 'subnet-06706d0609ad95781', and 'subnet-0f2323d033-309f'. The Security section lists 'VPC security groups' (default [sg-0ec24861fafd442e8]), 'Publicly accessible' (Yes), 'Certificate authority' (Info rds-e-2019), and 'Certificate authority date' (August 22, 2024, 18:08 (UTC+01:00)). The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray.

### 13. RDS security group has port 3306 opened for all incoming connections

The screenshot shows the AWS EC2 Security Groups console. A success message at the top states: "Inbound security group rules successfully modified on security group (sg-0ed24861fafd442e8 | default)". The main page displays the details of the security group "sg-0ed24861fafd442e8 - default". Key information includes:

- Security group name: default
- Security group ID: sg-0ed24861fafd442e8
- Description: default VPC security group
- VPC ID: vpc-0dc027bc3c9cfa5b4
- Owner: 380752321839
- Inbound rules count: 3 Permission entries
- Outbound rules count: 1 Permission entry

The "Inbound rules" tab is selected, showing three rules:

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sgr-0a66008779d6fe9	IPv4	MySQL/Aurora	TCP	3306	0.0.0.0/0	-
-	sgr-02e4f7cccd4c92df	-	All traffic	All	All	sg-0ed24861fafd442e...	-
-	sgr-02c03c71b09ed5e5	IPv4	All traffic	All	All	172.31.0.0/16	-

Step number B

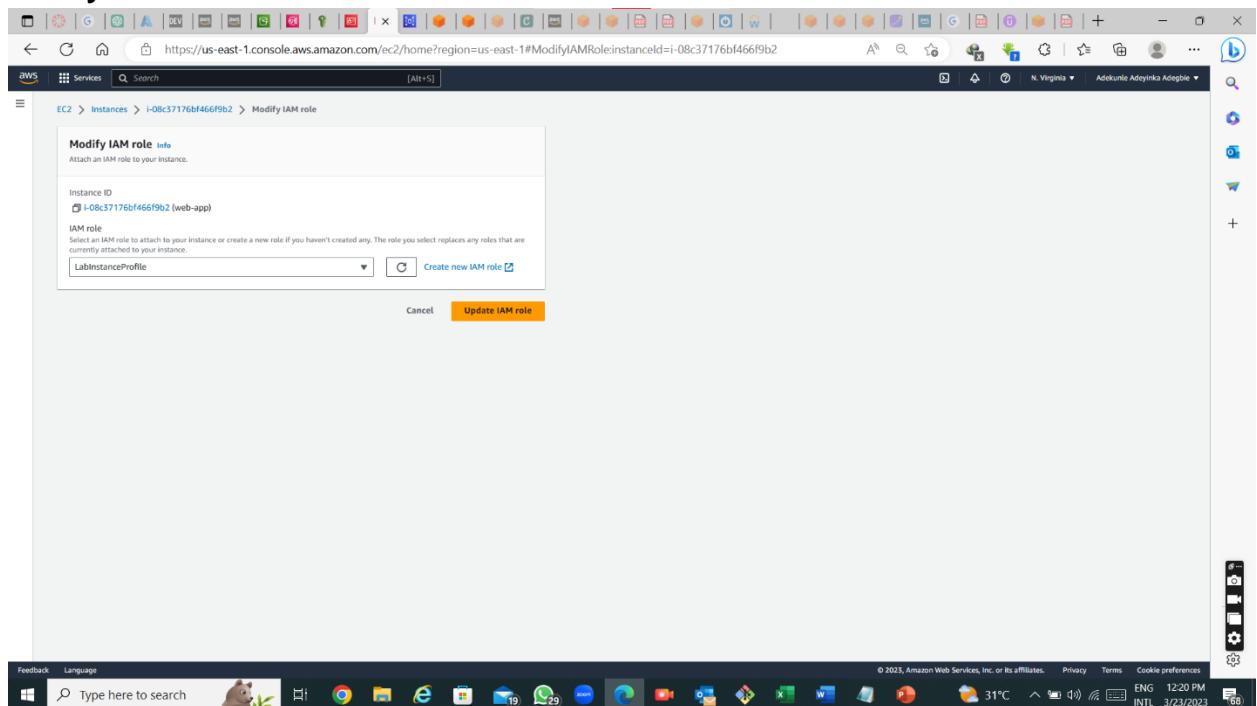
Step name Assignment of IAM role for EC2 instance

Instructions

- 1) Navigate back to EC2- > Instances
- 2) Select the EC2 instance created in the previous step and select Actions-> Security -> Modify IAM role
- 3) Select the role LabInstanceProfile from the dropdown and click on Save

Expected screenshots  
1) Modify IAM role screen

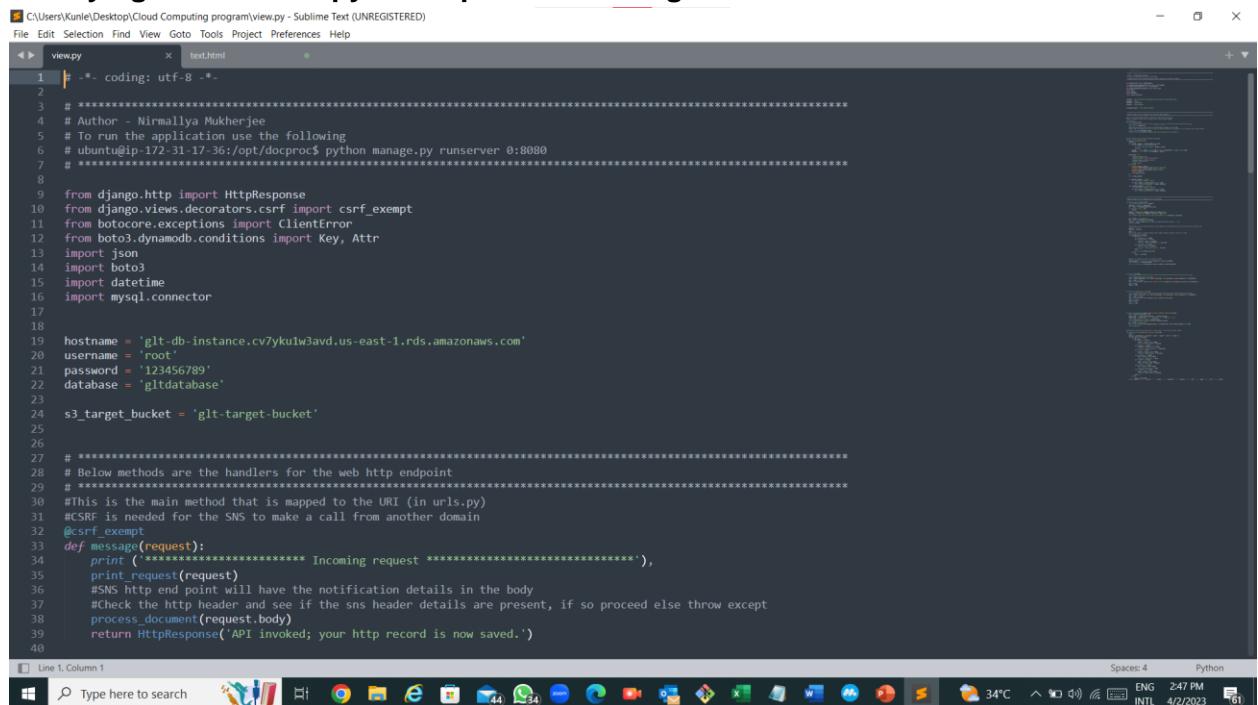
## 14 Modify IAM role screen



Step number	C
Step name	Configuration and Uploading of custom program
Instructions	<ol style="list-style-type: none"><li>1) Download the file <b>docproc-new.zip</b> on your machine</li><li>2) Unzip the downloaded file</li><li>3) Enter the unzipped folder and open the file <a href="#">views.py</a> in the API folder using a text editor</li><li>4) In line number 19-24, modify the target bucket name to the one created in Step 2 (a) and modify the hostname, username, password and database variables to the values set while creating the RDS database and save the file</li><li>5) Copy the folder docproc-new to the home folder of the EC2 instance created in Step 3(a) using scp. Use the command given below <code>scp -i &lt;pem&gt; -r ./docproc-new ec2-user@&lt;ip&gt;:/home/ec2-user</code></li></ol>

Expecte 1) Modifying of the [views.py](#) file to point to  
d the target bucket 2)Copying the folder to the  
screens EC2 instance  
hots

## 15 Modifying of the views.py file to point to the target bucket



The screenshot shows a Sublime Text window with two tabs: 'view.py' and 'tex.html'. The 'view.py' tab is active and contains Python code. The code imports various modules like django.http, django.views.decorators.csrf, boto3, json, datetime, and mysql.connector. It defines variables for hostname, username, password, and database, and sets s3\_target\_bucket to 'glt-target-bucket'. The code then defines a csrf\_exempted message function that prints incoming requests and processes them if they contain specific headers. The Sublime Text interface includes a sidebar with file navigation, a status bar at the bottom, and a taskbar with various application icons.

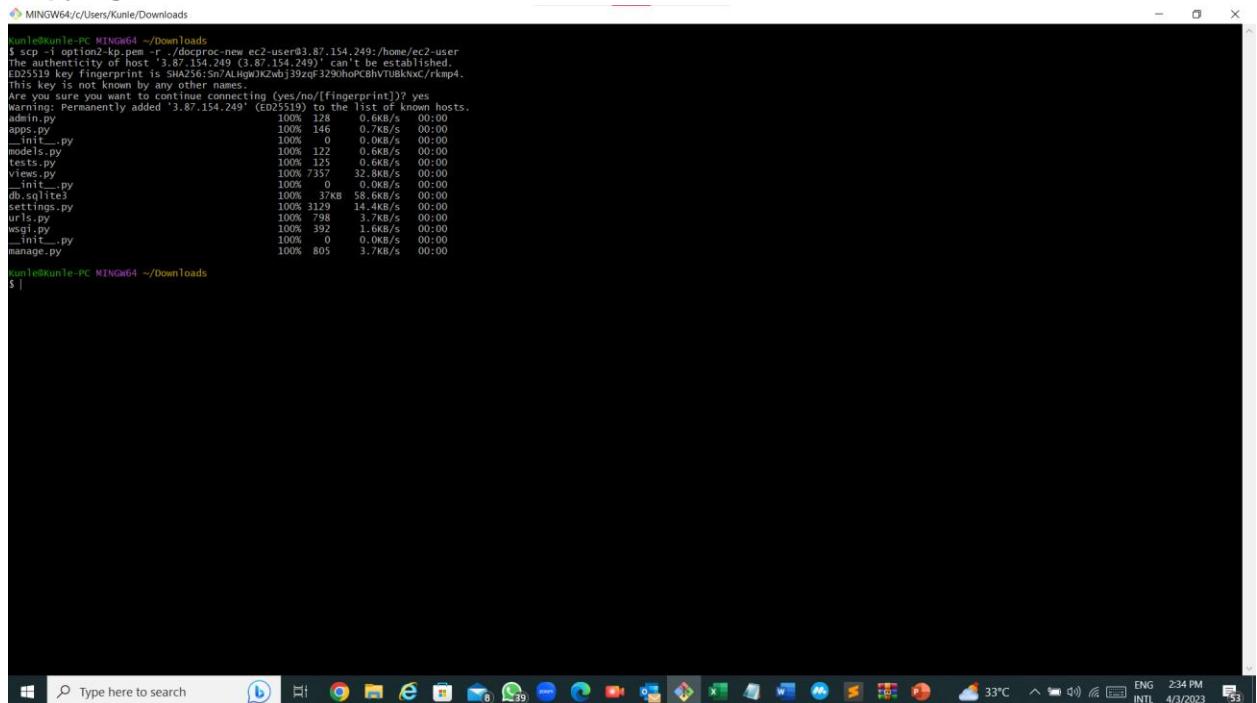
```

C:\Users\Kunle\Desktop\Cloud Computing program\view.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

view.py
1 | # -*- coding: utf-8 -*-
2 |
3 | # *****
4 | # Author - Nirmallya Mukherjee
5 | # To run the application use the following
6 | # ubuntu@ip-172-31-17-36:/opt/docproc$ python manage.py runserver 0:8080
7 | # *****
8 |
9 | from django.http import HttpResponseRedirect
10| from django.views.decorators.csrf import csrf_exempt
11| from botocore.exceptions import ClientError
12| from boto3.dynamodb.conditions import Key, Attr
13| import json
14| import boto3
15| import datetime
16| import mysql.connector
17|
18|
19| hostname = 'glt-db-instance.cv7yku1w3avd.us-east-1.rds.amazonaws.com'
20| username = 'root'
21| password = '123456789'
22| database = 'gltdatabase'
23|
24| s3_target_bucket = 'glt-target-bucket'
25|
26|
27| # Below methods are the handlers for the web http endpoint
28| # *****
29| # This is the main method that is mapped to the URI (in urls.py)
30| # CSRF is needed for the SNS to make a call from another domain
31| @csrf_exempt
32| def message(request):
33|     def message(request):
34|         print ('***** Incoming request *****')
35|         print request
36|         #SNS http end point will have the notification details in the body
37|         #Check the http header and see if the sns header details are present, if so proceed else throw exception
38|         process_document(request.body)
39|         return HttpResponseRedirect('API invoked; your http record is now saved.')
40|

```

## 16 Copying the folder to the EC2 Instance



```
lunileKunle-PC MINGW64 ~/Downloads
$ scp -i option2-kp.pem -r ./docproc-new ec2-user@3.87.154.249:/home/ec2-user
The authenticity of host '3.87.154.249 (3.87.154.249)' can't be established.
ED25519 key fingerprint is SHA256:uqkxWdJ39zqg329hohClnv10sBkxcJ/kMq4.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
warning: Permanently added '3.87.154.249' (ED25519) to the list of known hosts.
admin.py
    100% 128     0.6KB/s   0:00
__init__.py
    100% 146     0.7KB/s   0:00
models.py
    100% 0      0.0KB/s   0:00
tests.py
    100% 122     0.6KB/s   0:00
views.py
    100% 125     0.6KB/s   0:00
__init__.py
    100% 735     32.0KB/s  0:00
db.sqlite3
    100% 37KB   58.6KB/s   0:00
settings.py
    100% 3129    14.4KB/s  0:00
urls.py
    100% 798     3.1KB/s   0:00
wsgi.py
    100% 393     1.6KB/s   0:00
__init__.py
    100% 0      0.0KB/s   0:00
manage.py
    100% 805     3.7KB/s   0:00

lunileKunle-PC MINGW64 ~/Downloads
$ |
```

### Step 3: Creation and Verification of SNS subscription and Generation of CSV file

Step number a

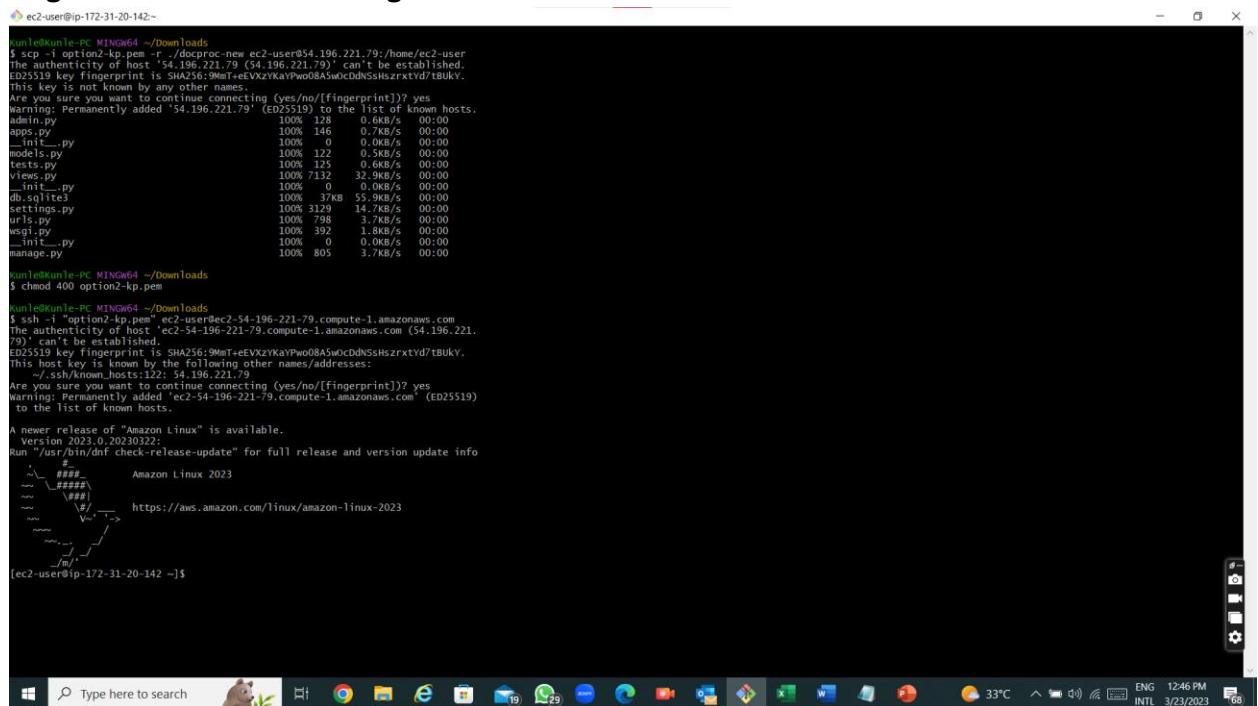
Step name Starting the  
EC2 custom  
program

Instructions 1) Log into the EC2 instance using SSH  
2) Run the following commands after successful SSH to start the server  
sudo cp -r docproc-new /opt  
sudo chown ec2-user:ec2-user -R /opt  
cd /opt/docproc-new  
sudo yum update  
sudo yum install python-pip -y  
python -m pip install --upgrade pip setuptools  
sudo pip install virtualenv  
virtualenv ~/.virtualenvs/djangodev  
source ~/.virtualenvs/djangodev/bin/activate  
pip install django  
pip install boto3  
pip install mysql-connector-python-rf  
python manage.py runserver 0:8080

**Keep this terminal window open throughout the rest of the exercise**

Expected      1) Server in  
screenshots    waiting state

## 17 Log into EC2 instance using SSH



The screenshot shows a Windows desktop environment with a terminal window open. The terminal window displays the following command-line session:

```
curlie@kunle-PC MINGW64 ~/Downloads
$ scp -i option2-kp.pem ec2-user@ec2-user@54.196.221.79:/home/ec2-user
The authenticity of host '54.196.221.79' (54.196.221.79) can't be established.
ED25519 key fingerprint is SHA256:9Mmt+eEVx2YKaYpwo08ASwcdCdn55Hs2rzYd7tBUKY.
This host key is not known by me. Are you sure you want to continue connecting (yes/no/(fingerprint))? yes
warning: Permanently added '54.196.221.79' (ED25519) to the list of known hosts.
admin.py
apps.py
auth.py
models.py
tests.py
views.py
__init__.py
db.sqlite3
settings.py
urls.py
wsgi.py
__init__.py
manage.py
curlie@kunle-PC MINGW64 ~/Downloads
$ chmod 400 option2-kp.pem
curlie@kunle-PC MINGW64 ~/Downloads
$ ssh -i "option2-kp.pem" ec2-user@ec2-54-196-221-79.compute-1.amazonaws.com
The authenticity of host 'ec2-54-196-221-79.compute-1.amazonaws.com' (54.196.221.79) can't be established.
ED25519 key fingerprint is SHA256:9Mmt+eEVx2YKaYpwo08ASwcdCdn55Hs2rzYd7tBUKY.
This host key is known by the following other names/addresses:
  ./ssh/known_hosts:122: 54.196.221.79
Are you sure you want to continue connecting (yes/no/(fingerprint))? yes
warning: Permanently added 'ec2-54-196-221-79.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

A newer release of "Amazon Linux" is available.
Version 2023.0.20230322:
Run "/usr/bin/dnf check-release-update" for full release and version update info
      _#
     /###
    /###_
   /#####
  /#####
 /#####
/#####
      Amazon Linux 2023
      _#
     /###
    /###_
   /#####
  /#####
 /#####
/#####
      https://aws.amazon.com/linux/amazon-linux-2023
[ec2-user@ip-172-31-20-142 ~]$
```

The terminal window is part of a desktop environment with a taskbar at the bottom containing various application icons.

## 18 Server in waiting state

```
ec2-user@ip-172-31-20-14:~/dproc-new
Support for Python 2.7 in January 2021. More details about Python 2 support in pip
can be found at https://pypi.pypy.io/en/latest/development/release-process/#python-2-support pip 21.0 will remove support for this functionality.
Collecting boto3
  Downloading boto3-1.17.112-py2.py3-none-any.whl (131 kB) | 131 kB 22.1 MB/s
Collecting jmespath==1.0.0,>=0.7.1
  Downloading jmespath-0.10.0-py2.py3-none-any.whl (24 kB)
Collecting s3transfer<0.5.0,>=0.4.0
  Downloading s3transfer-0.4.2-py2.py3-none-any.whl (79 kB) | 79 kB 9.6 MB/s
Collecting botocore<1.21.0,>=1.20.112
  Downloading botocore-1.20.112-py2.py3-none-any.whl (7.7 MB) | 7.7 MB 39.0 MB/s
Collecting futures<4.0.0,>=2.0; python_version == "2.7"
  Downloading futures-3.4.0-py2-none-any.whl (16 kB)
Collecting python-dateutil<3.0.0,>=2.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB) | 247 kB 47.5 MB/s
Collecting urllib3<1.27,*,>1.25.4
  Downloading urllib3-1.26.15-py2.py3-none-any.whl (140 kB) | 140 kB 52.8 MB/s
Collecting six<1.3
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: jmespath, futures, six, python-dateutil, urllib3, botocore, s3transfer, boto3
Successfully installed boto3-1.17.112 botocore-1.20.112 futures-3.4.0 jmespath-0.10.0 python-dateutil-2.8.2 s3transfer-0.4.2 six-1.16.0 urllib3-1.26.15
(django-pdev) [ec2-user@ip-172-31-20-14 dproc-new]$ pip install mysql-connector-python-rf
DEPRECATION: Python 2.7 reached the end of its life on January 1st, 2020. Please
upgrade your Python as Python 2.7 is no longer maintained. pip 21.0 will drop s
upport for Python 2.7 in January 2021. More details about Python 2 support in pi
p can be found at https://pypi.pypy.io/en/latest/development/release-process/#pyt
hon-2-support pip 21.0 will remove support for this functionality.
Collecting mysql-connector-python-rf
  Downloading mysql-connector-python-rf-2.2.2.tar.gz (11.9 kB) | 11.9 kB 26.3 MB/s
Building wheels for collected packages: mysql-connector-python-rf
  Building wheel for mysql-connector-python-rf (setup.py) ... done
    Created wheel for mysql-connector-python-rf: filename=mysql_connector_python_rf-2.2.2-cp27-cp27mu-linux_x86_64.whl size=249459 sha256=4601ad2dad2f58a914f7b6161419a1a434c10d88e6a3a4f3ed
    Stored in directory: /var/folders/0v/0vqjwzqj7qg444f3k.../pip/wheels/3b/b5/d4/5d0e3338625186a
b2fbf75908b5817bb59aa1ef1d291a0fa
Successfully built mysql-connector-python-rf
Installing collected packages: mysql-connector-python-rf
Successfully installed mysql-connector-python-rf-2.2.2
(django-pdev) [ec2-user@ip-172-31-20-14 dproc-new]$ python manage.py runserver 0:8080
Performing system checks...
System check identified no issues (0 silenced).
April 01, 2023 - 13:46:38
Django version 1.11.29, using settings 'dproc.settings'
Starting development server at http://0:8080/
Quit the server with CONTROL-C.
```

Step number b

Step name	Creation of SNS subscription
Instructions	<ol style="list-style-type: none"><li>1) Navigate to SNS in the AWS Console and select the topic S3ToEC2Topic</li><li>2) Click on Create Subscription</li><li>3) Enter the following details Protocol : HTTP Endpoint : <code>http://&lt;host&gt;:8080/sns</code> where &lt;host&gt; in the public IP of the EC2 instance Click on Create Subscription</li><li>4) In the EC2 terminal window, look for the field "SubscribeURL" and copy the entire link given</li></ol> <p><b>Note: If a message is seen "ValueError: No JSON object could be decoded", it can be safely ignored</b></p> <ol style="list-style-type: none"><li>5) Paste that link into a browser window to verify the SNS subscription (Ignore any messages received in the web browser)</li></ol>
Expected screenshots	1) Subscription URL in EC2 terminal window

## 19. Subscription URL in EC2 terminal window (i)

```
ec2-user@ip-172-31-20-14:~/opt/docproc-new
K0Wg9MLBt5diHPKAhblLTXMSZcuIFCkxwM92TfEmi96dLXet6wV/fal.wMFn0dB1J1j2wTR
tkjGhsow70bt/AwAr3/p4jyxa585omsw/bf lyBw="
"SigningCertURL" : "https://sns.us-east-1.amazonaws.com/SimpleNotificationServ
ice-56e67fc4b41ffec09b0196692625d385.pem"
}
Request method = POST
The S3 JSON is [
  {
    "type" : "Subscriptionconfirmation",
    "MessageId" : "cebe1bd2-a106-4b29-938b-1fed2058d883",
    "Token" : "2336412f7fb687f5ds1fe6425c464de128861f50fdf1fa9a288ab3a8e5d57581
6ffcf0f419459bc2d00f42bd6a11c94b2bd3c62e5f1e6db898bc38341af6b2952e19c99bf
81e602f7da5a2e1c283d004903fa7faa157d0079c46272ef74612243b18adab30351fc868f7",
    "TopicArn" : "arn:aws:sns:us-east-1:380732321839:S3t0c27topic",
    "Message" : "You have chosen to subscribe to the topic arn:aws:sns:us-east-1:3
80732321839:S3t0c27topic. To confirm the subscription, visit the SubscribeURL i
ncluded in this message."
  }
]
SubscribeURL : "https://sns.us-east-1.amazonaws.com/?Action=ConfirmSubscribe
&TopicArn=arn:aws:sns:us-east-1:380732321839:S3t0c27topic&Token=2336412f7fb6
8ffcf0f419459bc2d00f42bd6a11c94b2bd3c62e5f1e6db898bc38341af6b2952e19c99bf
81e602f7da5a2e1c283d004903fa7faa157d0079c46272ef74612243b18adab30351fc868f7",
"SignatureVersion" : "2",
"Signature" : "x5N8Rq9-660uA32Pwx2x5LXvx-x7Wt8v0hKMVs2v1+2TPr/UEG31hynePVzB
BPgjLGjxuspvZbg1Rwp9XCRU1wmxexy26drLYmf7svxvfbY7/tkCRT/HQj0j78sszAve8
9KC61vhdsz+E047imh0PLXG5rc0o+mQcudc8Ht88bHVw23Ys1eihv1hmzuwVsuzmo/ypvFw9
K0Wg9MLBt5diHPKAhblLTXMSZcuIFCkxwM92TfEmi96dLXet6wV/fal.wMFn0dB1J1j2wTR
tkjGhsow70bt/AwAr3/p4jyxa585omsw/bf lyBw="
"SigningCertURL" : "https://sns.us-east-1.amazonaws.com/SimpleNotificationServ
ice-56e67fc4b41ffec09b0196692625d385.pem"
}
Internal Server Error: /sns
Traceback (most recent call last):
  File "/home/ec2-user/.virtualenvs/djangodev/lib/python2.7/site-packages/django
/core/handlers/exception.py", line 41, in inner
    response = get_response(request)
  File "/home/ec2-user/.virtualenvs/djangodev/lib/python2.7/site-packages/django
/core/handlers/base.py", line 187, in get_response
    response = self.process_exception_by_middleware(e, request)
  File "/home/ec2-user/.virtualenvs/djangodev/lib/python2.7/site-packages/django
/core/handlers/base.py", line 185, in get_response
    response = wrapped_callback(request, *callback_args, **callback_kwargs)
  File "/home/ec2-user/.virtualenvs/djangodev/lib/python2.7/site-packages/django
/views/decorators/csrf.py", line 58, in wrapped_view
    return view_func(*args, **kwargs)
  File "/opt/docproc-new/api/views.py", line 39, in message
    process_document(request, body)
  File "/opt/docproc-new/api/views.py", line 86, in process_document
    s3 = json.loads(json.loads(Message))
  File "/usr/lib64/python2.7/json/_init__.py", line 339, in loads
    return _default_decoder.decode(s)
  File "/usr/lib64/python2.7/json/_decoder.py", line 364, in decode
    obj, end = self.raw_decode(s, idx=_w(s, 0, end))
  File "/usr/lib64/python2.7/json/_decoder.py", line 382, in raw_decode
    raise ValueError("No JSON object could be decoded")
ValueError: No JSON object could be decoded
[03/Apr/2023 13:50:08] "POST /sns HTTP/1.1" 500 87033
```

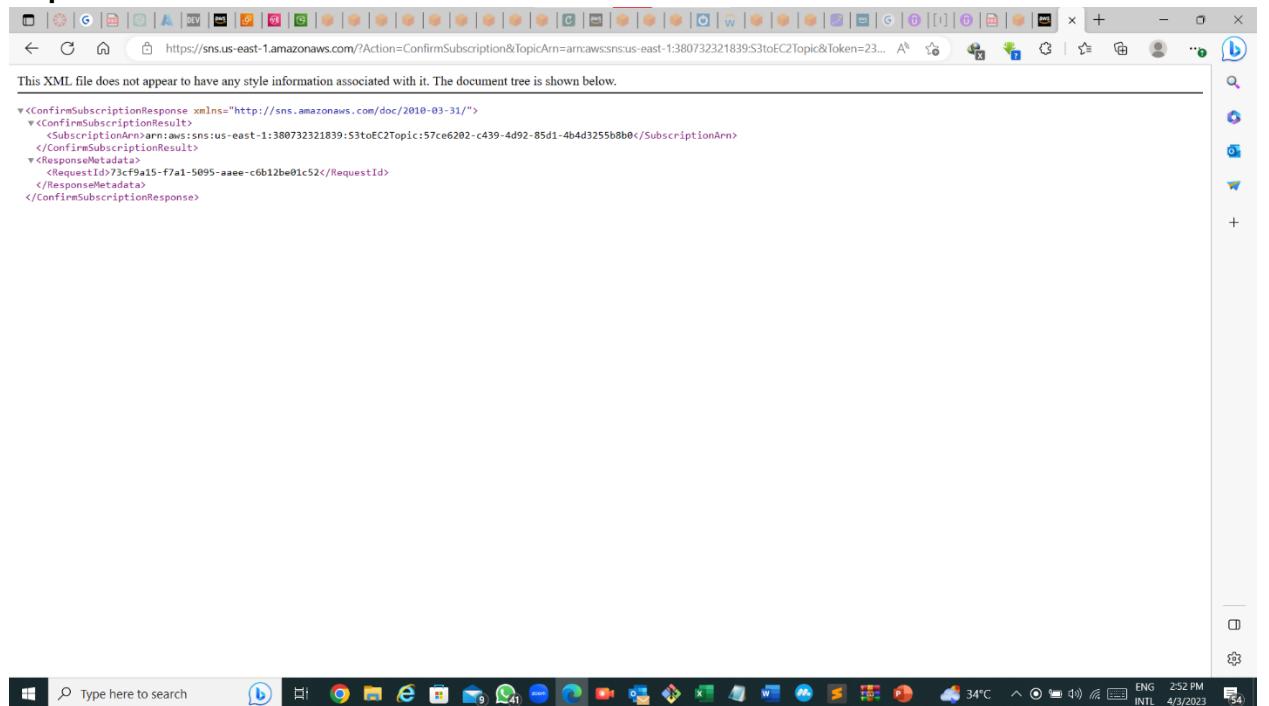


## 20. Subscription URL in EC2 terminal window (ii)

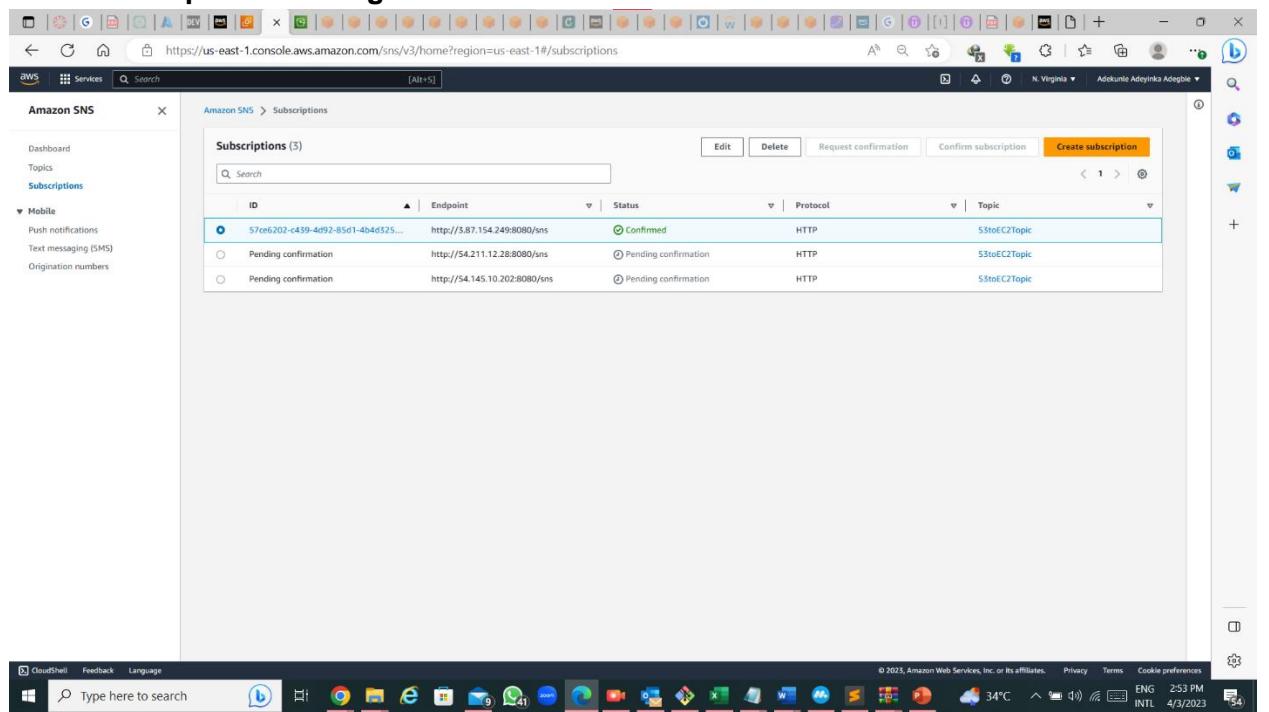
```
ec2-user@ip-172-31-20-14:~/opt/docproc-new
K0Wg9MLBt5diHPKAhblLTXMSZcuIFCkxwM92TfEmi96dLXet6wV/fal.wMFn0dB1J1j2wTR
tkjGhsow70bt/AwAr3/p4jyxa585omsw/bf lyBw="
"SigningCertURL" : "https://sns.us-east-1.amazonaws.com/SimpleNotificationServ
ice-56e67fc4b41ffec09b0196692625d385.pem"
}
Request method = POST
The S3 JSON is [
  {
    "type" : "Subscriptionconfirmation",
    "MessageId" : "cebe1bd2-a106-4b29-938b-1fed2058d883",
    "Token" : "2336412f7fb687f5ds1fe6425c464de128861f50fdf1fa9a288ab3a8e5d57581
6ffcf0f419459bc2d00f42bd6a11c94b2bd3c62e5f1e6db898bc38341af6b2952e19c99bf
81e602f7da5a2e1c283d004903fa7faa157d0079c46272ef74612243b18adab30351fc868f7",
    "TopicArn" : "arn:aws:sns:us-east-1:380732321839:S3t0c27topic",
    "Message" : "You have chosen to subscribe to the topic arn:aws:sns:us-east-1:3
80732321839:S3t0c27topic. To confirm the subscription, visit the SubscribeURL i
ncluded in this message."
  }
]
SubscribeURL : "https://sns.us-east-1.amazonaws.com/?Action=ConfirmSubscribe
&TopicArn=arn:aws:sns:us-east-1:380732321839:S3t0c27topic&Token=2336412f7fb6
8ffcf0f419459bc2d00f42bd6a11c94b2bd3c62e5f1e6db898bc38341af6b2952e19c99bf
81e602f7da5a2e1c283d004903fa7faa157d0079c46272ef74612243b18adab30351fc868f7",
"SignatureVersion" : "2",
"Signature" : "x5N8Rq9-660uA32Pwx2x5LXvx-x7Wt8v0hKMVs2v1+2TPr/UEG31hynePVzB
BPgjLGjxuspvZbg1Rwp9XCRU1wmxexy26drLYmf7svxvfbY7/tkCRT/HQj0j78sszAve8
9KC61vhdsz+E047imh0PLXG5rc0o+mQcudc8Ht88bHVw23Ys1eihv1hmzuwVsuzmo/ypvFw9
K0Wg9MLBt5diHPKAhblLTXMSZcuIFCkxwM92TfEmi96dLXet6wV/fal.wMFn0dB1J1j2wTR
tkjGhsow70bt/AwAr3/p4jyxa585omsw/bf lyBw="
"SigningCertURL" : "https://sns.us-east-1.amazonaws.com/SimpleNotificationServ
ice-56e67fc4b41ffec09b0196692625d385.pem"
}
Internal Server Error: /sns
Traceback (most recent call last):
  File "/home/ec2-user/.virtualenvs/djangodev/lib/python2.7/site-packages/django
/core/handlers/exception.py", line 41, in inner
    response = get_response(request)
  File "/home/ec2-user/.virtualenvs/djangodev/lib/python2.7/site-packages/django
/core/handlers/base.py", line 187, in get_response
    response = self.process_exception_by_middleware(e, request)
  File "/home/ec2-user/.virtualenvs/djangodev/lib/python2.7/site-packages/django
/core/handlers/base.py", line 185, in get_response
    response = wrapped_callback(request, *callback_args, **callback_kwargs)
  File "/home/ec2-user/.virtualenvs/djangodev/lib/python2.7/site-packages/django
/views/decorators/csrf.py", line 58, in wrapped_view
    return view_func(*args, **kwargs)
  File "/opt/docproc-new/api/views.py", line 39, in message
    process_document(request, body)
  File "/opt/docproc-new/api/views.py", line 86, in process_document
    s3 = json.loads(json.loads(Message))
  File "/usr/lib64/python2.7/json/_init__.py", line 339, in loads
    return _default_decoder.decode(s)
  File "/usr/lib64/python2.7/json/_decoder.py", line 364, in decode
    obj, end = self.raw_decode(s, idx=_w(s, 0, end))
  File "/usr/lib64/python2.7/json/_decoder.py", line 382, in raw_decode
    raise ValueError("No JSON object could be decoded")
ValueError: No JSON object could be decoded
[03/Apr/2023 13:50:08] "POST /sns HTTP/1.1" 500 87033
```



## 21. Copied SubscribeURL link in a browser



## 22. SNS Subscription showing confirmed status



Step number	c
Step name	Generation of CSV file
Instructions	<ol style="list-style-type: none"> <li>1) Download the file <b>docproc-invoice.txt</b> provided with this workbook</li> <li>2) Navigate to S3 in the AWS Console</li> <li>3) Upload the sample invoice file to the source S3 bucket using the default options</li> <li>4) Verify that a CSV file is generated in the target S3 bucket. This may take a few minutes</li> <li>5) (Optional) Login to the RDS instance using your preferred MySQL client and check the table created inside the specified database.</li> </ol>
Expected screenshots	<ol style="list-style-type: none"> <li>1) Generated CSV file in the target S3 bucket</li> </ol>

### 23. Docproc-invoice.txt file uploaded into the “glt-source-bucket”

The screenshot shows the AWS S3 console interface. At the top, there's a navigation bar with various icons and a search bar. Below it, a banner displays a green success message: "Upload succeeded" with a link to "View details below.". A modal window titled "Upload: status" is open, containing a summary table and a detailed file list.

Summary	
Destination s3://glt-source-bucket	Succeeded 1 file, 572.0 B (100.00%)
	Failed 0 files, 0 B (0%)

Below the summary, there are tabs for "Files and folders" (which is selected) and "Configuration". Under "Files and folders", there's a table showing one file:

Files and folders (1 Total, 572.0 B)						
Find by name						
Name	Folder	Type	Size	Status	Error	
docproc-invoice.txt	-	text/plain	572.0 B	Succeeded	-	

At the bottom of the screen, there's a Windows taskbar with various pinned icons and a system tray showing the date and time.

## 24. Generated CSV file in “glt-target-bucket”

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with 'Amazon S3' selected under 'Buckets'. The main area shows the 'glt-target-bucket' with one object listed: 'Objects (1)'. The object is 'docproc-invoice.csv', which is a CSV file. The file details show it was last modified on April 3, 2023, at 15:06:07 (UTC+01:00), has a size of 611.0 B, and is stored in the Standard storage class.

### Answer the following questions

Q Which of the following properties of an AWS resource is sufficient and necessary to uniquely identify it across all of AWS?

- a) ARN
- b) Region and ARN
- c) ARN and Account number
- d) Depends on the resource used

Enter your answer here

A

Q Which of the following step numbers in Step 1 allowed S3 to publish to the SNS topic created?

2

- a) 1(a)
- b) 1(c)
- c) 1(d)
- d) 1(b)

Enter your answer here

D

Q Which port is being used by SNS to send the notification to the custom program?

3

- a) 8081
- b) 80
- c) 8080
- d) 8065

Enter your answer here

B

Q How many IAM roles can be attached to an EC2 instance at a time?

4

- a) 2
- b) 3
- c) 1
- d) Depends on the policies required

Enter your answer here

C

Q As a product manager, how would you describe the benefits of this architecture to an client, as compared to 5 an equivalent on-premises architecture?

As a product manager, the following are some benefits that I would highlight to a client about the benefits of using AWS SNS, RDS, and S3 over an on-premises architecture. This can enable such clients better meet both the needs of business and customers and stay ahead of competition:

1. **Scalability:** With AWS SNS, RDS, and S3, client can easily scale up or down their infrastructure based on the demands of their business, without having to invest in additional hardware or worry about capacity planning. This means that they can respond quickly to changes in their business needs and avoid over-provisioning or under-provisioning your infrastructure.
2. **Cost-effectiveness:** By using AWS SNS, RDS, and S3, client can avoid the upfront costs associated with purchasing and maintaining hardware, and instead pay only for the resources they use. This means that client can reduce their capital expenditure and shift to an operational expenditure model, which can improve their cash flow and help them manage costs more effectively.
3. **Resilience and high availability:** AWS SNS, RDS, and S3 provide available and fault-tolerant infrastructure that is designed to minimize downtime and provide reliable access to data. With AWS SNS, client can ensure that they messages are delivered reliably to their subscribers, even in the event of a failure in one of their systems. With AWS RDS, they can leverage automatic failover and backup and recovery features to ensure that their databases are always available and protected. With AWS S3, they can replicate their data across multiple availability zones within a region, providing automatic failover in case of a failure in one zone.

4. **Security and compliance:** AWS SNS, RDS, and S3 provide a wide range of security and compliance features to help client protect their data, including built-in encryption, identity and access management, and compliance certifications. AWS also provides them with visibility and control over your infrastructure, which can help you meet your regulatory requirements and improve your overall security posture.
5. **Innovation and agility:** AWS SNS, RDS, and S3 provide a wide range of services and tools that enable client to innovate quickly and bring new products and features to market faster. With AWS, they can take advantage of the latest technologies and best practices without having to worry about managing the underlying infrastructure.

Grades distribution	
MCQs	10 (2.5 mark each)
Subjective questions	6 marks
Implementation screenshots	24 marks (2 marks each)
Total	40 marks