

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

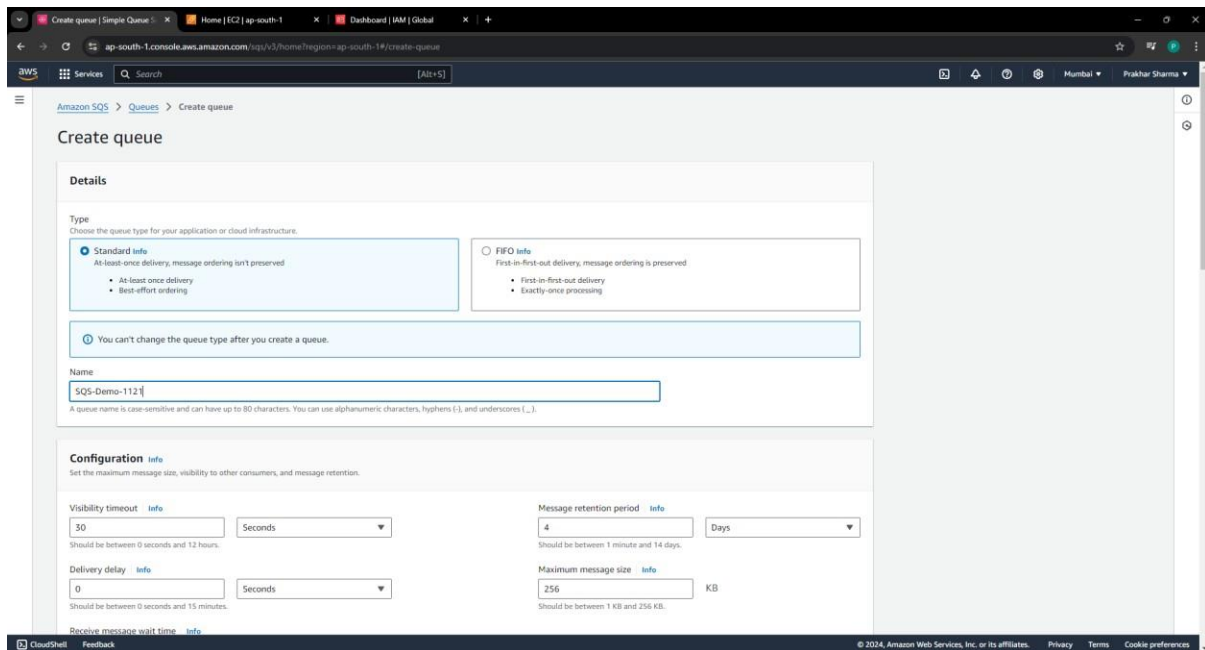
Name of the Student: **Prakhar Anil Sharma**

PRN: **20220801121**

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**

1. Create an SQS Queue

1. Open the AWS Console, search for SQS, and click on Simple Queue Service (SQS).
2. Click on **Create Queue**.
3. Give a name to your queue.



The screenshot shows the AWS Console 'Create queue' page. The 'Details' section has two options: 'Standard' (selected) and 'FIFO'. The 'Name' field is filled with 'SQS-Demo-112'. The 'Configuration' section has four settings: 'Visibility timeout' (30 seconds), 'Delivery delay' (0 seconds), 'Message retention period' (4 days), and 'Maximum message size' (256 KB). The 'Create queue' button is visible at the bottom right.

4. Scroll down and click **Create Queue**.

School of Computer Science, Engineering and Applications(SCSEA)

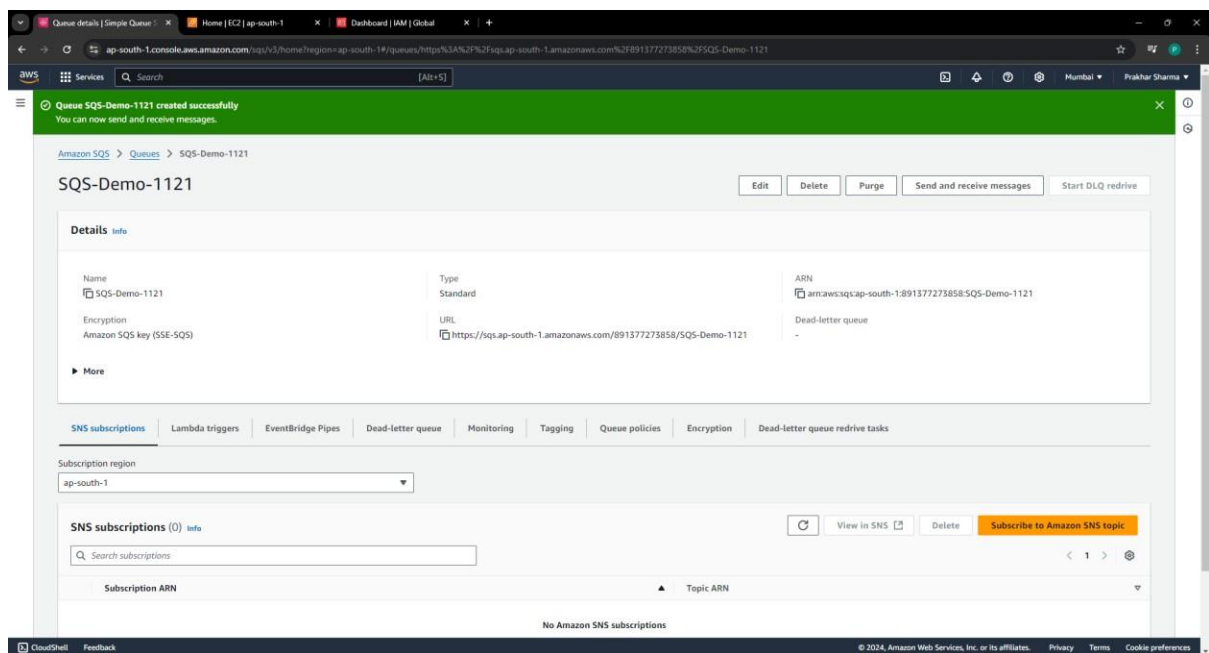
B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: Prakhar Anil Sharma

PRN: 20220801121

**Title of Practical: AWS Simple Queue Service (SQS) For Sending
Messages**



1. Launch an EC2 Instance

1. Search for EC2 and click on **Launch Instance**.
2. Give a name to your EC2 instance.
 - For AMI, select Ubuntu, and in the dropdown, select Ubuntu 22.04.

School of Computer Science, Engineering and Applications(SCSEA)

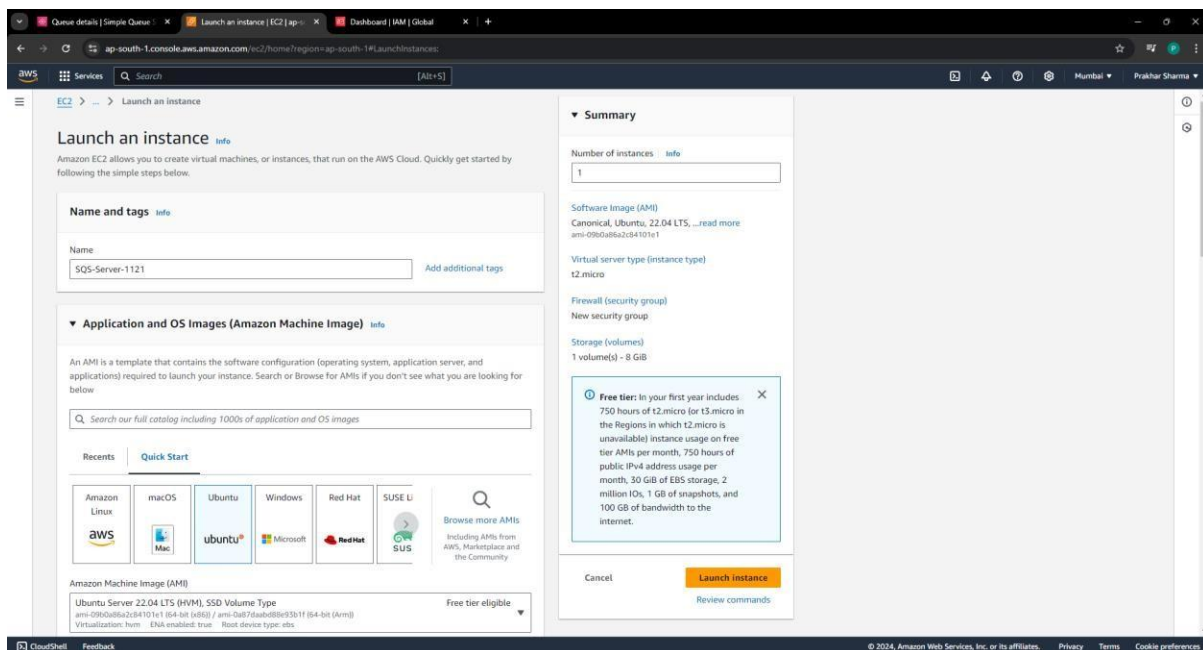
B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

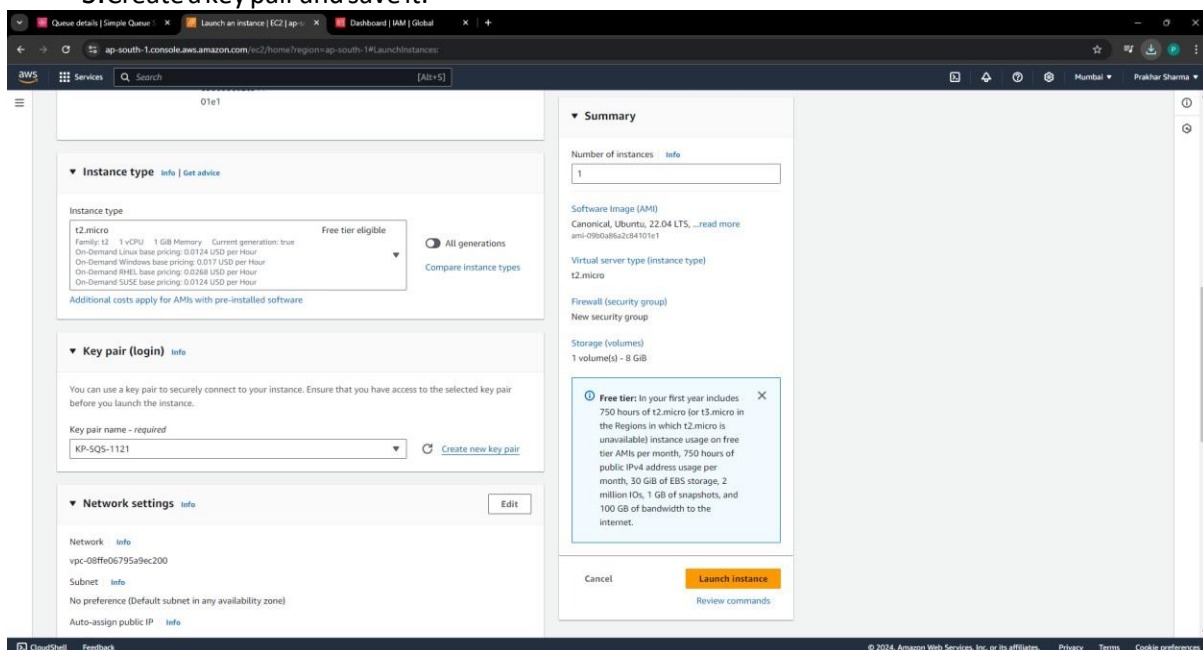
Name of the Student: **Prakhar Anil Sharma**

PRN: **20220801121**

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**



3. Create a key pair and save it.



School of Computer Science, Engineering and Applications(SCSEA)
B.C.A. TY (CCSA)
Subject : Advanced Cloud Computing (P)

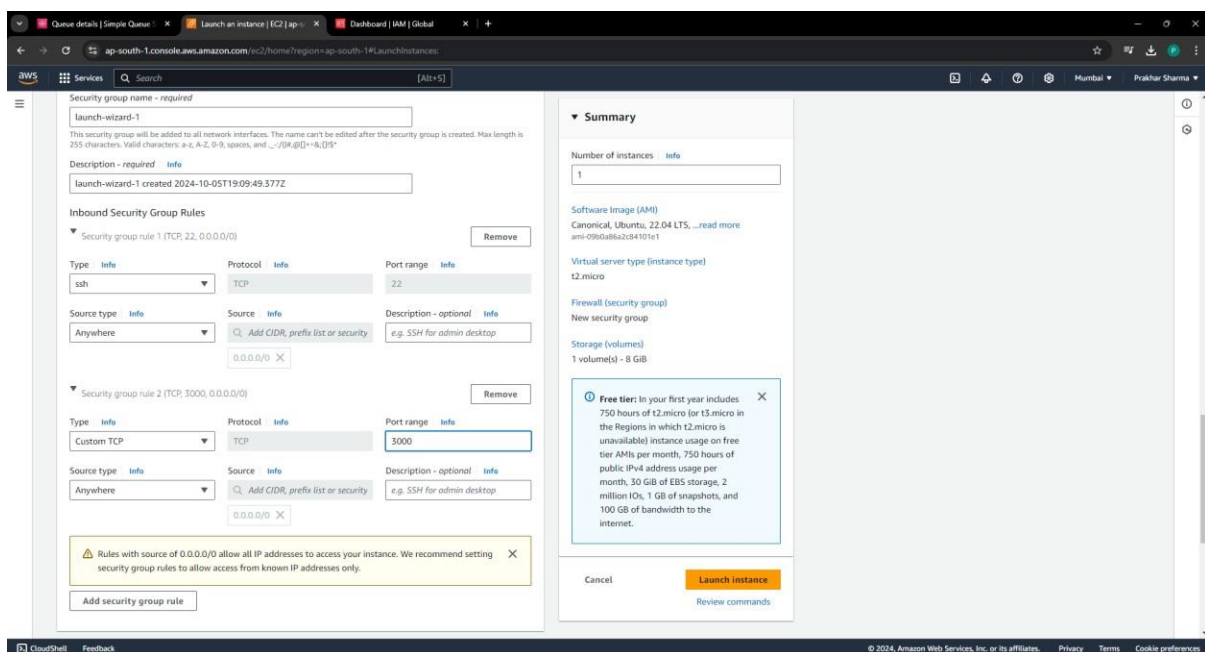
Name of the Student: **Prakhar Anil Sharma**

PRN: 20220801121

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**

4. In the Network section, click on **Edit** and select **Auto-assign IP**. Enable it and click **Add**.

5. Click on **Add Security Group Rule** in the Inbound Security Group Rules subsection of network settings.



6. Click on **Launch Instance**.

School of Computer Science, Engineering and Applications(SCSEA)

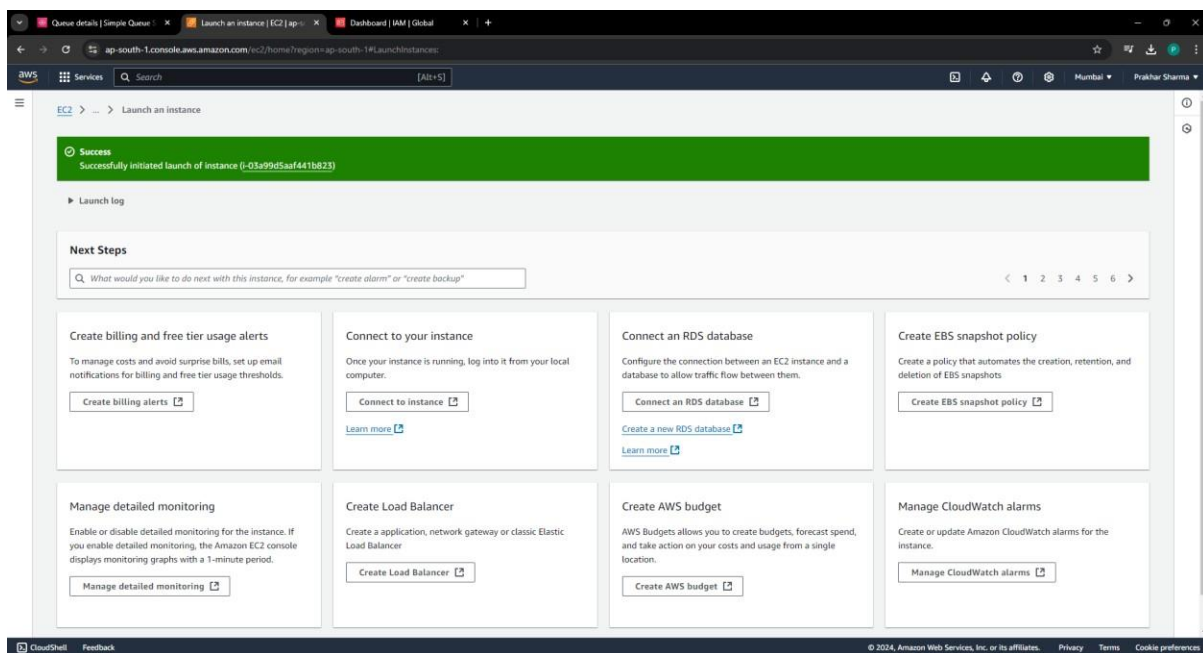
B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: **Prakhar Anil Sharma**

PRN: 20220801121

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**



2. Create an IAM Role

1. Search for IAM (Identity Access Management).
2. In IAM, on the left-side panel, click on **Roles**.
3. Click on **Create Role**.
4. In the **Select trusted entity** section, choose **AWS Service** and select **EC2** in the use case.

School of Computer Science, Engineering and Applications(SCSEA)

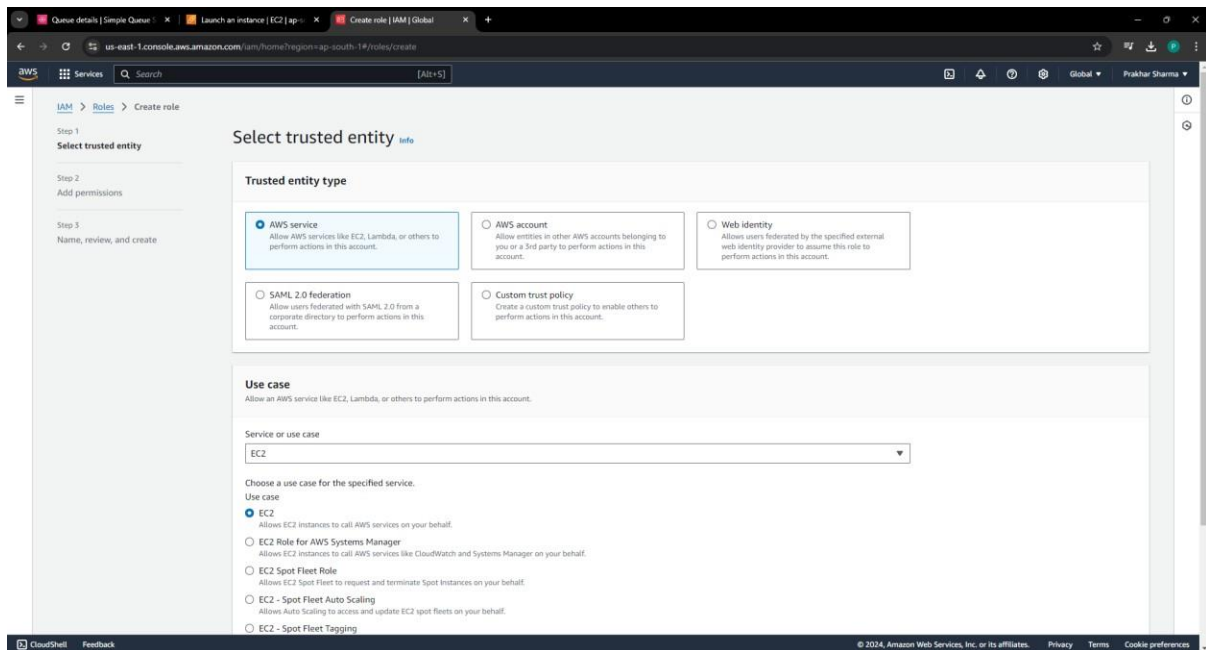
B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

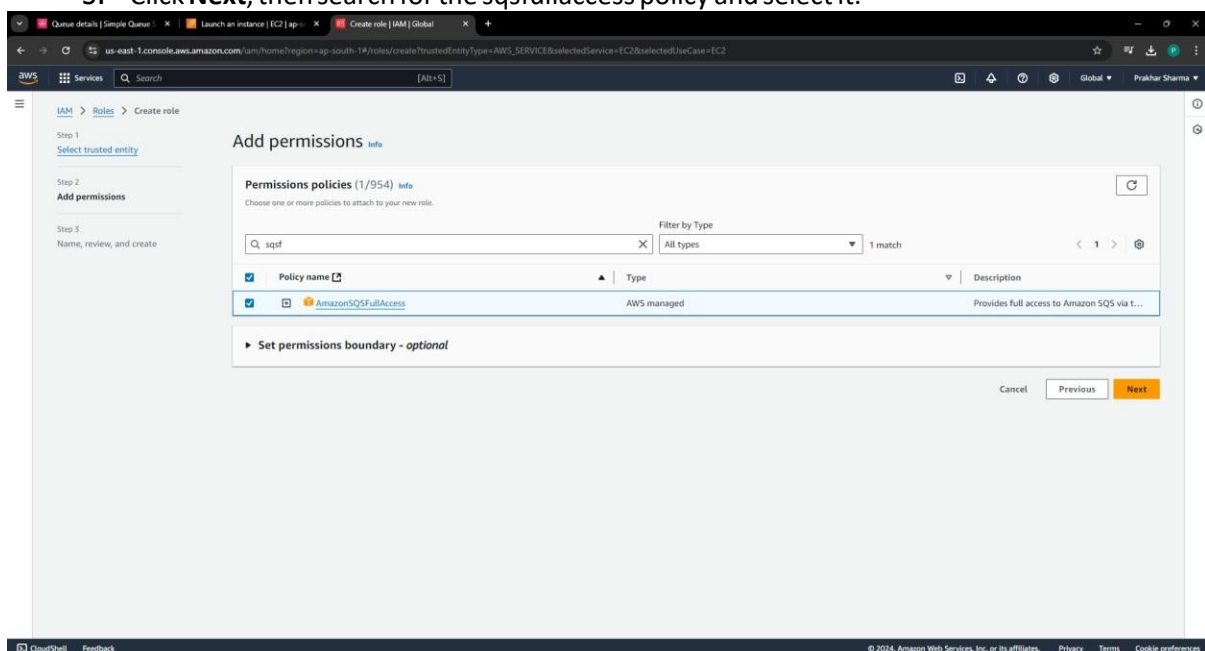
Name of the Student: **Prakhar Anil Sharma**

PRN: **20220801121**

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**



5. Click **Next**, then search for the **sqsfullaccess** policy and select it.



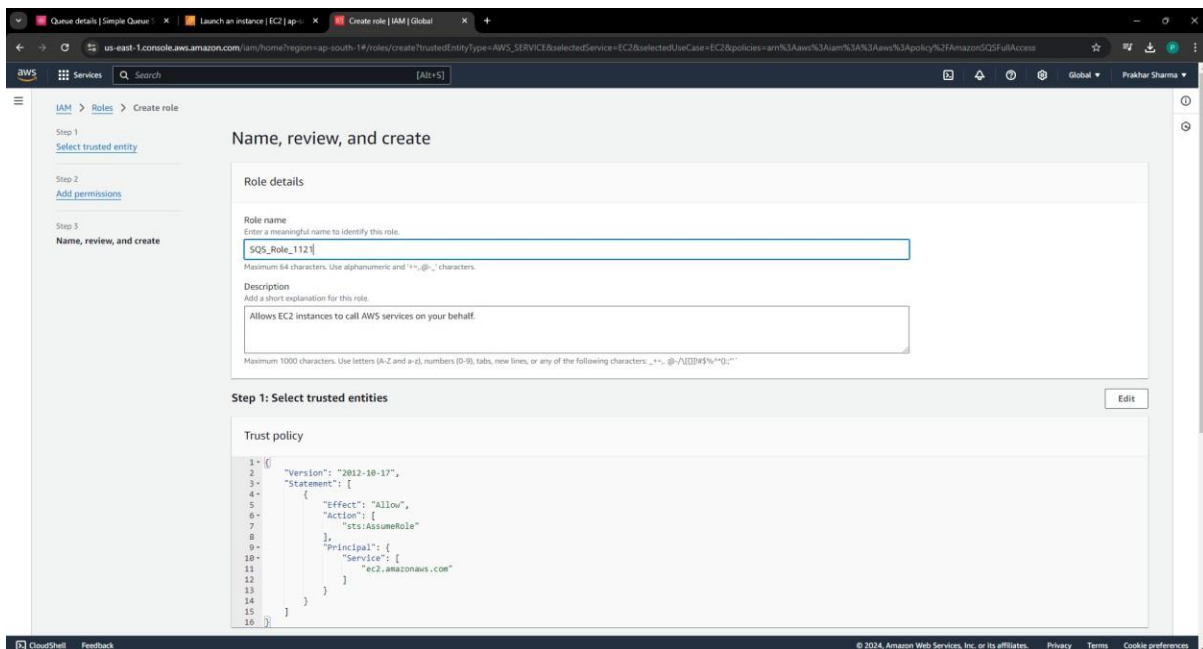
School of Computer Science, Engineering and Applications(SCSEA)
B.C.A. TY (CCSA)
Subject : Advanced Cloud Computing (P)

Name of the Student: **Prakhar Anil Sharma**

PRN: 20220801121

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**

6. Click **Next**, give a name to the role, and click **Create Role**.



The screenshot shows the AWS IAM console 'Create role' page. The 'Role name' field is filled with 'SQS_Role_1121'. The 'Description' field contains 'Allows EC2 instances to call AWS services on your behalf.' Below this is the 'Step 1: Select trusted entities' section, which shows a 'Trust policy' code block. The code block contains a JSON policy that allows the role to assume the role of 'ec2.amazonaws.com'.

7. After creating the role successfully, navigate back to your previously created EC2 instance.

8. Click on **Actions**, select **Security**, then **Modify IAM**

Role.

9. Select the role you just created.

School of Computer Science, Engineering and Applications(SCSEA)

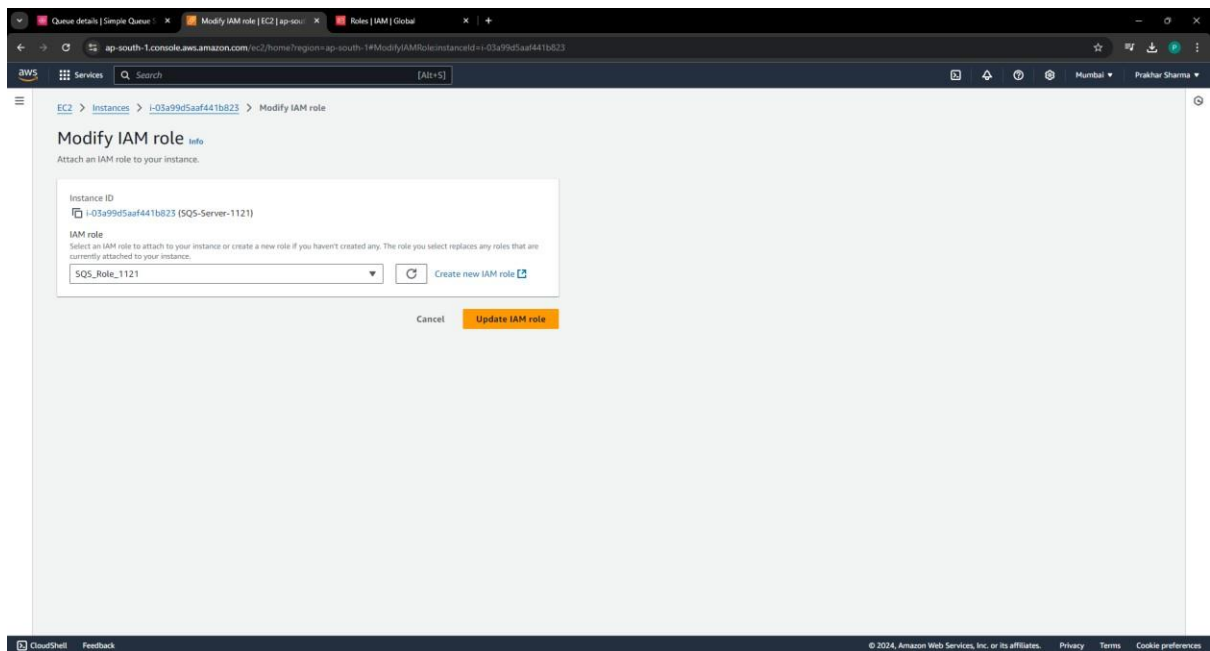
B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

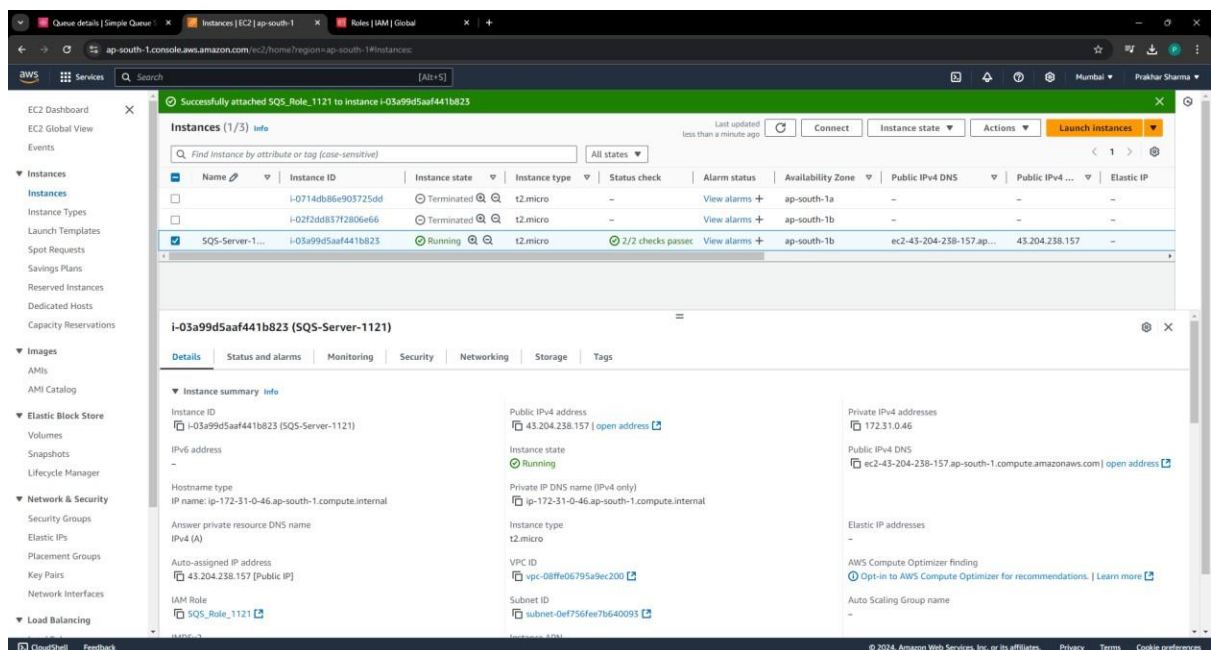
Name of the Student: **Prakhar Anil Sharma**

PRN: 20220801121

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**



10. Click Update IAM Role.



PRN: 20220801121

8

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

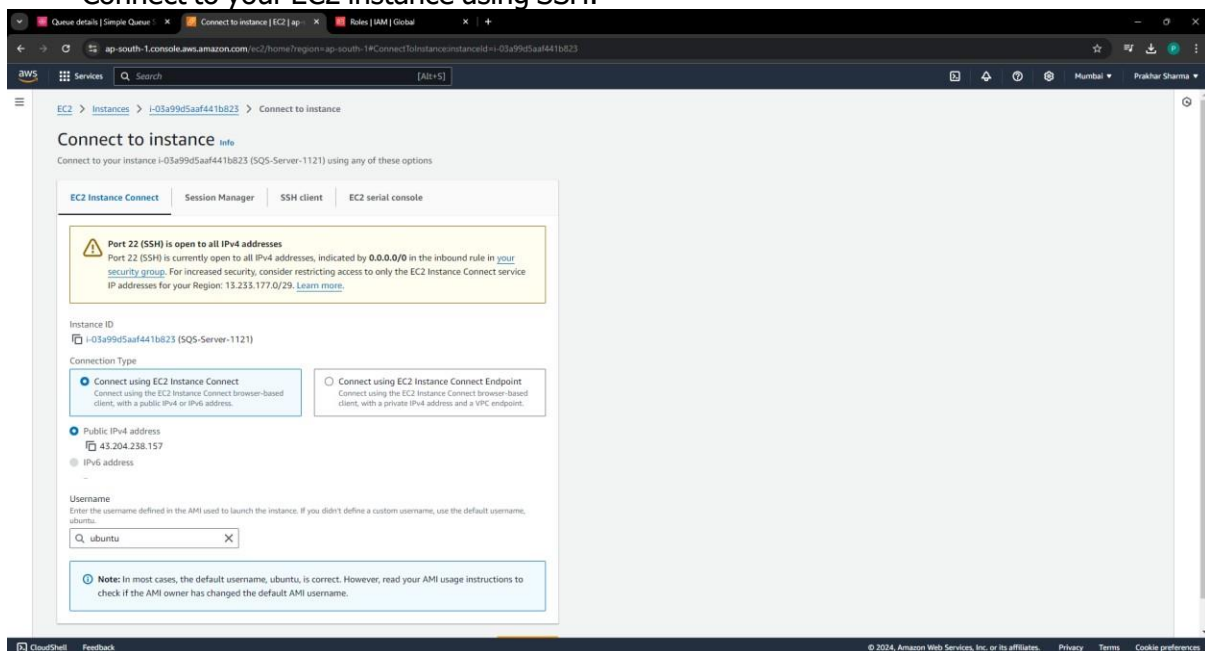
Name of the Student: **Prakhar Anil Sharma**

PRN: 20220801121

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**

4. Connect to the EC2 Instance

Connect to your EC2 instance using SSH.



5. Update and Set Up the EC2 Instance

5.1. Update and Upgrade Your Machine, and Install Node.js and npm

After connecting to your EC2 instance, run the following commands:

```
sudo apt update && sudo apt upgrade -y && sudo apt install nodejs -y && sudo  
apt install npm -y
```

School of Computer Science, Engineering and Applications(SCSEA)

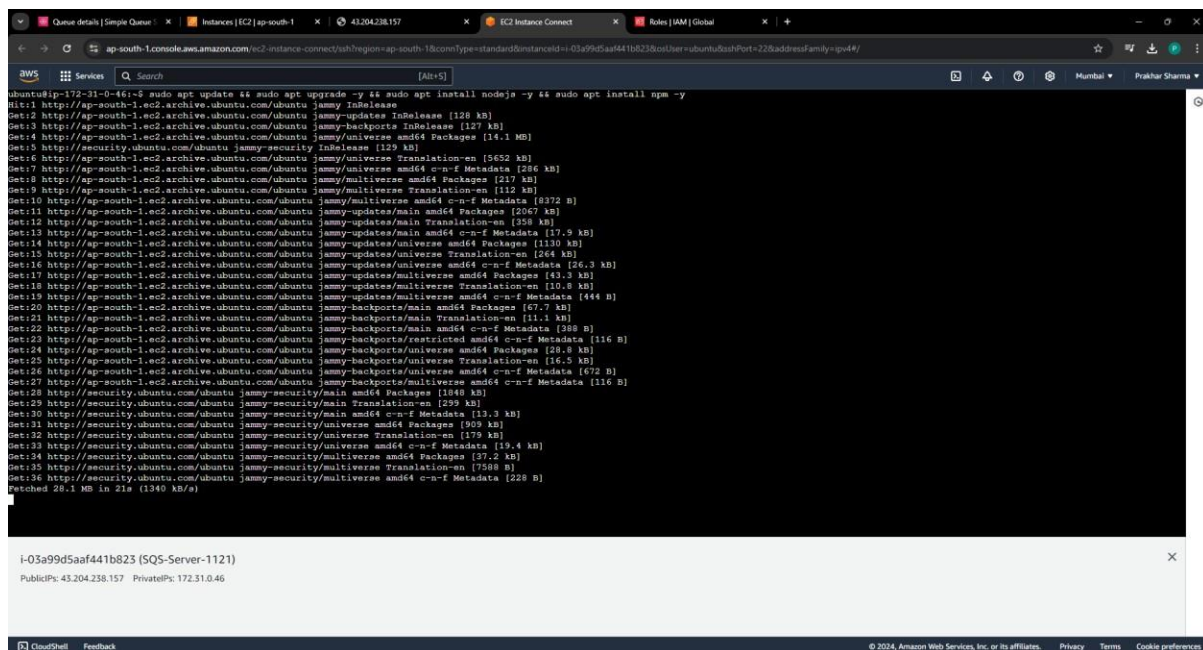
B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: Prakhar Anil Sharma

PRN: 20220801121

Title of Practical: AWS Simple Queue Service (SQS) For Sending Messages



```
ubuntu@ip-172-31-0-46:~$ sudo apt update && sudo apt upgrade -y && sudo apt install nodejs -y && sudo apt install npm -y
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:5 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:6 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:7 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:8 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:9 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:10 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:11 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [2067 kB]
Get:12 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [358 kB]
Get:13 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [17.9 kB]
Get:14 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1130 kB]
Get:15 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [264 kB]
Get:16 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [26.3 kB]
Get:17 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [43.3 kB]
Get:18 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [10.8 kB]
Get:19 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [444 B]
Get:20 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [67.7 kB]
Get:21 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [11.1 kB]
Get:22 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 c-n-f Metadata [388 B]
Get:23 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/restricted amd64 c-n-f Metadata [116 B]
Get:24 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [28.8 kB]
Get:25 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe Translation-en [16.5 kB]
Get:26 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 c-n-f Metadata [672 B]
Get:27 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:28 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1848 kB]
Get:29 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [299 kB]
Get:30 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [13.3 kB]
Get:31 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [209 kB]
Get:32 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [179 kB]
Get:33 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 c-n-f Metadata [19.4 kB]
Get:34 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [37.2 kB]
Get:35 http://security.ubuntu.com/ubuntu jammy-security/multiverse Translation-en [7588 B]
Get:36 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 c-n-f Metadata [228 B]
Fetched 28.1 MB in 21s (1340 kB/s)
```

5.3 Create app.js and Add Content

1) Create the app.js

file:

nano ~/app.js

PRN: 20220801121

School of Computer Science, Engineering and Applications(SCSEA)

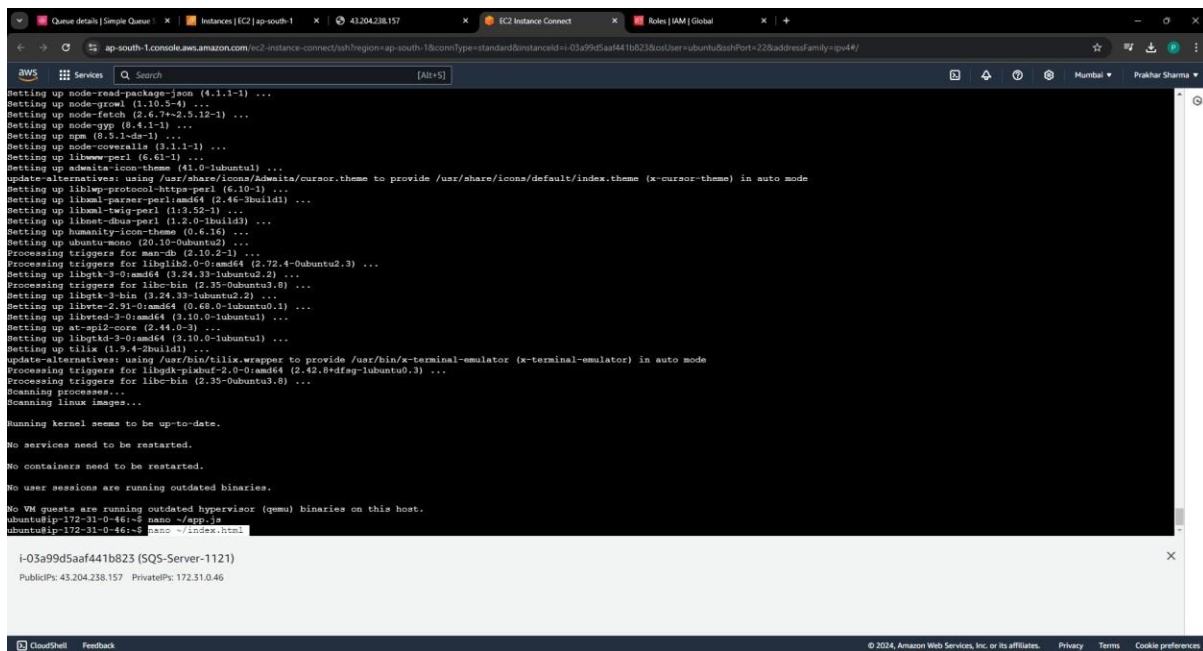
B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: **Prakhar Anil Sharma**

PRN: **20220801121**

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**



```
Setting up node-read-package-json (4.1.1-1) ...
Setting up node-gyp (8.4.1-1) ...
Setting up node-cvss (3.1.1-1) ...
Setting up node-coveralls (3.1.1-1) ...
Setting up libwww-perl (6.61-1) ...
Setting up adwaita-icon-theme (41.0-1ubuntu1) ...
update-alternatives: using /usr/share/icons/Adwaita/cursor.theme to provide /usr/share/icons/default/index.theme (x-cursor-theme) in auto mode
Setting up libxml-perl (2.24-3) ...
Setting up libxml-twig-perl (1:3.52-1) ...
Setting up libnet-dbus-perl (1.2.0-1build3) ...
Setting up humanity-icon-theme (0.6.16) ...
Setting up ubuntu-mono (20.10-0ubuntu2) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libbluetooth2 (2.72.4-0ubuntu2.3) ...
Setting up libgtk-3-0:amd64 (3.24.33-1ubuntu2.2) ...
Processing triggers for libc-bin (2.35-0ubuntu3.8) ...
Setting up libgtk-3-bin (3.24.33-1ubuntu2.2) ...
Setting up libvte-2.91-0:amd64 (0.68.0-1ubuntu0.1) ...
Setting up libvte2-0:amd64 (0.10.0-1ubuntu1) ...
Setting up at-spi2-core (2.44.0-3) ...
Setting up libgtk-3-0:amd64 (3.10.0-1ubuntu1) ...
Setting up tilix (1.9.4-2build1) ...
update-alternatives: using /usr/bin/tilix.wrapper to provide /usr/bin/x-terminal-emulator (x-terminal-emulator) in auto mode
Processing triggers for libgdk-pixbuf2.0-0:amd64 (2.42.8+dfsg-1ubuntu0.3) ...
Processing triggers for libc-bin (2.35-0ubuntu3.8) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.
No services need to be restarted.
No containers need to be restarted.
No user sessions are running outdated binaries.
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-0-46:~$ nano /app.js
ubuntu@ip-172-31-0-46:~$ nano /index.html

i-03a99d5aaf441b823 [SQS-Server-1121]
PublicIP: 43.204.238.157 PrivateIP: 172.31.0.46
```

2. Copy and paste the following content into app.js. Replace the placeholder SQS URL with your actual SQS queue URL:

```
const express =
require('express');const
AWS = require('aws-
sdk');
const bodyParser =
require('body-parser');const ejs
= require('ejs');
const path = require('path');
const app =
```

PRN: 20220801121



D Y PATIL
INTERNATIONAL
UNIVERSITY
AKURDI PUNE

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: Prakhar Anil Sharma

PRN: 20220801121

Title of Practical: AWS Simple Queue Service (SQS) For Sending Messages

```
express();  
  
const port =  
3000;  
  
  
// Configure AWS SDK to use IAM role credentials automatically  
AWS.config.update({ region: 'ap-south-1' }); // Replace with your desired  
AWS region  
  
  
// Create SQS service object  
const sqs = new AWS.SQS({ apiVersion: '2012-11-05' });  
  
  
// Body parser middleware  
app.use(bodyParser.urlencoded({ extended: true  
}));  
  
// Set EJS as the view engine and set the views  
directoryapp.set('view engine', 'ejs');  
app.set('views', path.join(__dirname, 'views'));  
  
  
// Serve  
index.html  
app.get('/',  
(req, res) => {  
  res.sendFile(path.join(__dirname, 'index.html'));
```

PRN: 20220801121

1
2



D Y PATIL
INTERNATIONAL
UNIVERSITY
AKURDI PUNE

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: Prakhar Anil Sharma

PRN: 20220801121

Title of Practical: AWS Simple Queue Service (SQS) For Sending Messages

```
});

// Serve send.html
app.get('/send', (req,
res) => {
  res.sendFile(path.join(__dirname, 'send.html'));
});

// Send message to
SQS app.post('/send',
(req, res) => {const {
message } = req.body;

const params
= {
  MessageBody
  : message,
  QueueUrl: 'https://sqs.ap-south-1.amazonaws.com/123123123/yourqueue', //
Replace with your SQS queue URL
});

sqs.sendMessage(params, (err,
data) => {if (err) {
```

PRN: 20220801121

1
3



D Y PATIL
INTERNATIONAL
UNIVERSITY
AKURDI PUNE

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: Prakhar Anil Sharma

PRN: 20220801121

Title of Practical: AWS Simple Queue Service (SQS) For Sending Messages

```
        console.error('Error sending message to SQS:',  
        err); res.status(500).send('Error sending  
        message to SQS');  
    } else {  
        console.log('Message sent to SQS:',  
        data.MessageId);res.redirect('/');  
    }  
}  
}  
);  
;  
  
// Serve messages.ejs  
app.get('/messages', (req,  
res) => {const params =  
{  
    QueueUrl: 'https://sqs.ap-south-1.amazonaws.com/123123123/yourqueue', //  
    Replace with your SQS queue URL  
    AttributeNames: ['All'],  
    MaxNumberOfMessages: 10, // Adjust as  
    neededWaitTimeSeconds: 0,  
};  
  
sqs.receiveMessage(params, (err,  
data) => {if (err) {  
    console.error('Error receiving messages from SQS:', err);
```

PRN: 20220801121

1
4



School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: **Prakhar Anil Sharma**

PRN: **20220801121**

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**

```
res.status(500).send('Error receiving messages from SQS');
} else {
  const messages =
    data.Messages || [];
  res.render('messages', {
    messages });
}
});
});

// Listen
on port
app.listen(
port, () =>
{
  console.log(`Server is running at http://localhost:${port}`);
});
```

3. Save and close the file:

- Press Ctrl + O to save the file.
- Press Enter to confirm.
- Press Ctrl + X to exit the editor.

5.2. Create index.html and Add Content



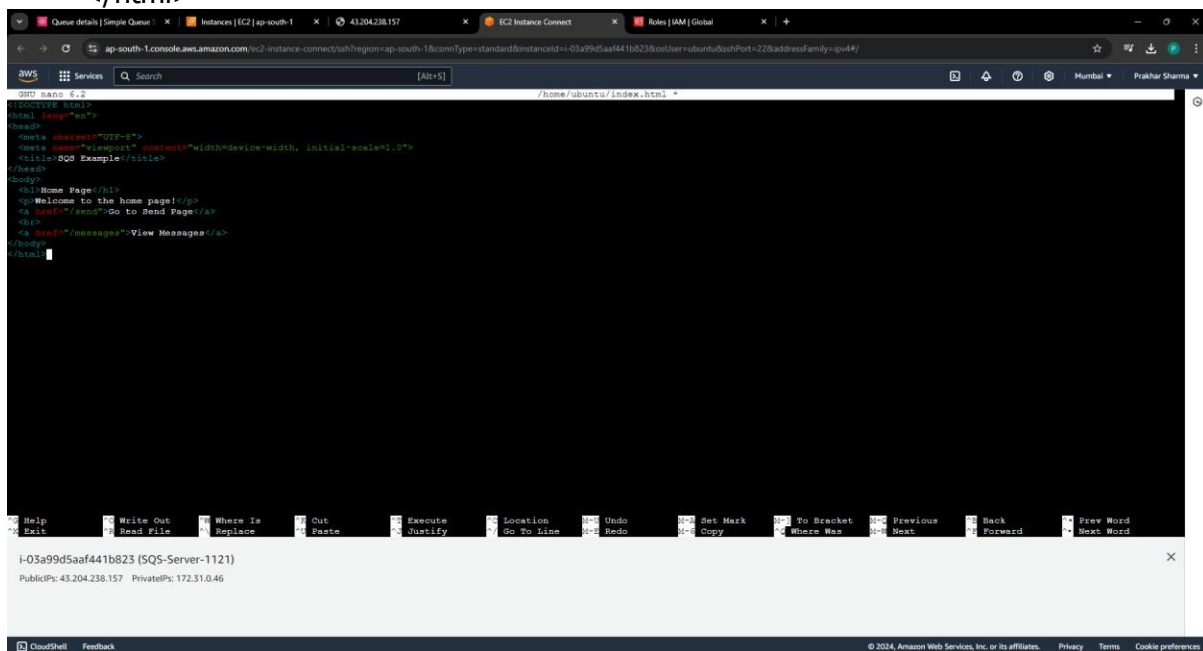
School of Computer Science, Engineering and Applications(SCSEA)
B.C.A. TY (CCSA)
Subject : Advanced Cloud Computing (P)

Name of the Student: **Prakhar Anil Sharma**

PRN: **20220801121**

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**

```
<p>Welcome to the home page!</p>
<a href="/send">Go to Send Page</a>
<br>
<a href="/messages">View Messages</a>
</body>
</html>
```



3. Save and close the file:

- Press Ctrl + O to save the file.
- Press Enter to confirm.
- Press Ctrl + X to exit the editor.

5.3. Create send.html and Add Content

1. Create the

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: **Prakhar Anil Sharma**

PRN: 20220801121

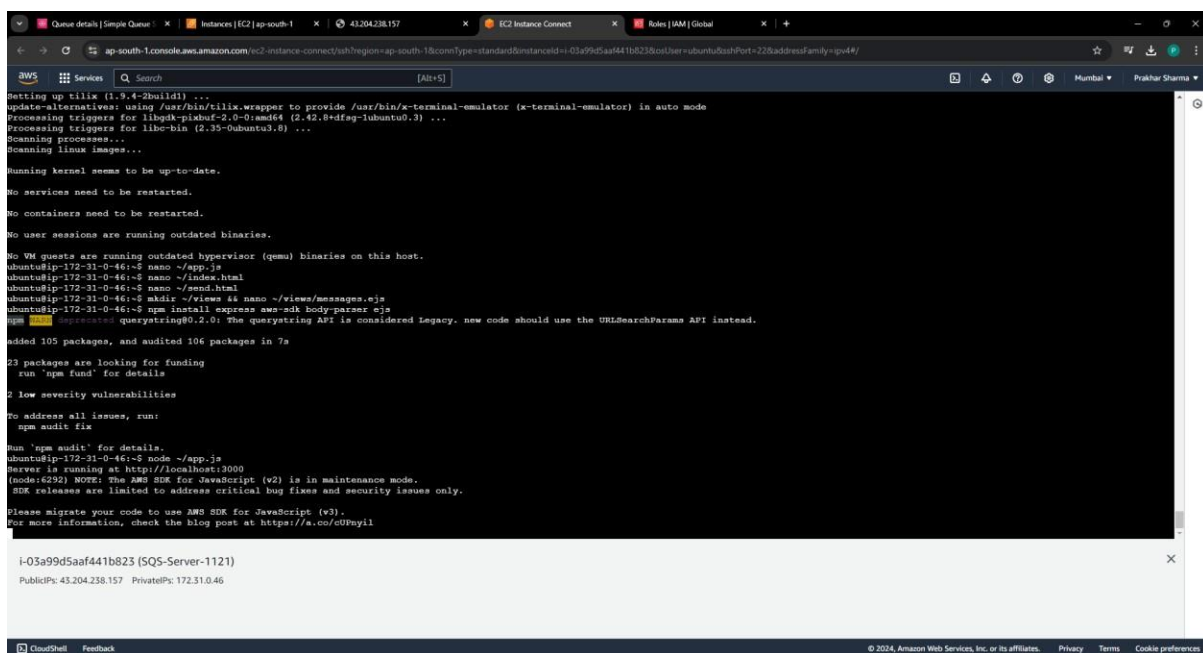
Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**

- Press Ctrl + O to save the file.
- Press Enter to confirm.
- Press Ctrl + X to exit the editor.

5.4. Create the views Directory and messages.ejs File

1. Create the views directory and the messages.ejs file inside the views

directory:mkdir ~/views && nano ~/views/messages.ejs



```
Setting up tilt (1.9.4-2build1) ...
update-alternatives: using /usr/bin/tilt.wrapper to provide /usr/bin/x-terminal-emulator (x-terminal-emulator) in auto mode
Processing triggers for libgdk-pixbuf2.0-0:amd64 (2.42.0+dfsg-1ubuntu0.3) ...
Processing triggers for libc-bin (2.35-0ubuntu3.8) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-0-46:~$ nano ~/app.js
ubuntu@ip-172-31-0-46:~$ nano ~/index.html
ubuntu@ip-172-31-0-46:~$ nano ~/send.html
ubuntu@ip-172-31-0-46:~$ mkdir ~/views && nano ~/views/messages.ejs
ubuntu@ip-172-31-0-46:~$ npm install express aws-sdk body-parser ejs
npm WARN deprecated querystring@0.2.0: The querystring API is considered Legacy. new code should use the URLSearchParams API instead.
added 105 packages, and audited 106 packages in 7s
23 packages are looking for funding
  run `npm fund` for details
2 low severity vulnerabilities
To address all issues, run:
  npm audit fix
Run `npm audit` for details.
ubuntu@ip-172-31-0-46:~$ node ~/app.js
Server is running at http://localhost:3000
(node:6292) NOTE: The AWS SDK for JavaScript (v2) is in maintenance mode.
SDK releases are limited to address critical bug fixes and security issues only.
Please migrate your code to use AWS SDK for JavaScript (v3).
For more information, check the blog post at https://a.co/cUmyil

i-03a99d5aaf441b823 [SQS-Server-1121]
PublicIP: 45.204.238.157 PrivateIP: 172.31.0.46
```

2. Copy and paste the following content into messages.ejs:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

PRN: 20220801121



D Y PATIL
INTERNATIONAL
UNIVERSITY
AKURDI PUNE

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: Prakhar Anil Sharma

PRN: 20220801121

**Title of Practical: AWS Simple Queue Service (SQS) For Sending
Messages**

```
<title>SQS Example</title>
</head>
<body>
<h1>Messages from SQS Queue</h1>
<ul>
<% messages.forEach(message => { %>
  <li><%= message.Body %></li>
<% }); %>
</ul>
<br>
<a href="/">Go to Home Page</a>
</body>
```

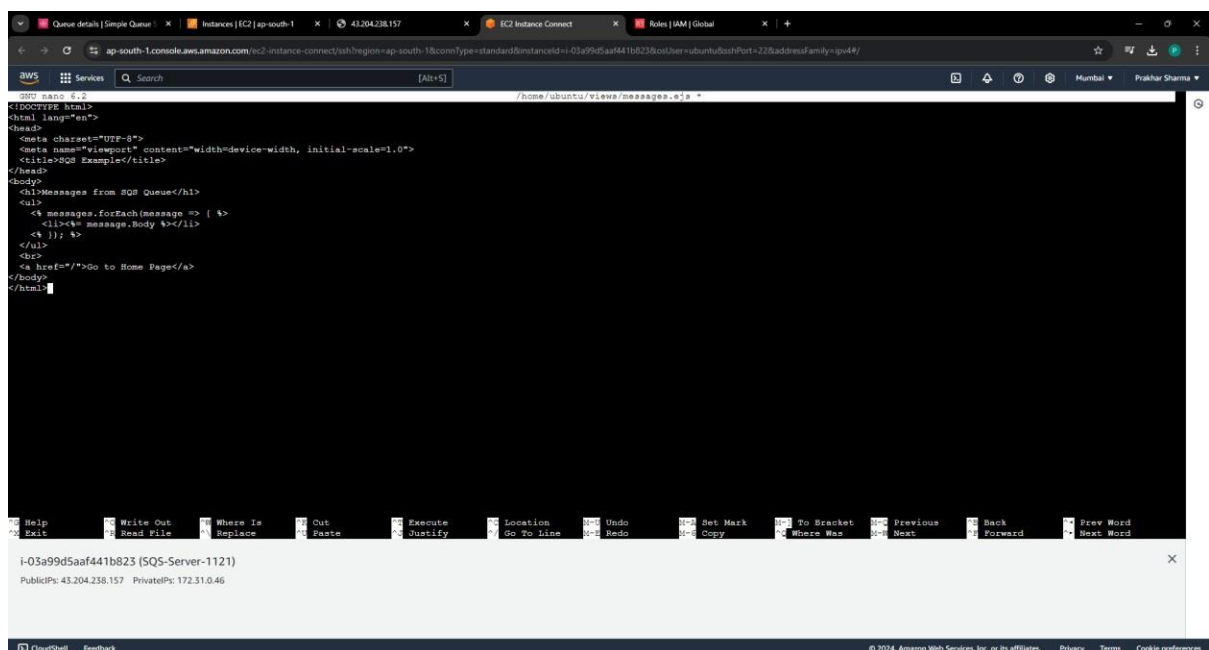
School of Computer Science, Engineering and Applications(SCSEA)
B.C.A. TY (CCSA)
Subject : Advanced Cloud Computing (P)

Name of the Student: **Prakhar Anil Sharma**

PRN: 20220801121

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**

</html>



```
2022 nano: 6.2 /home/ubuntu/view/messages.js
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>SQS Example</title>
</head>
<body>
  <h1>Messages from SQS Queue</h1>
  <ul>
    <li><%= messages.forEach(message => { %>
      <li><%= message.Body %></li>
    } %>
  </ul>
  <br>
  <a href="/">Go to Home Page</a>
</body>
</html>
```

3. Save and close the file:

- Press Ctrl + O to save the file.
- Press Enter to confirm.
- Press Ctrl + X to exit the editor.

5.5. Install Required Packages

Install the necessary Node.js packages:

npm install express aws-sdk body-parser ejs

5.6. Start the Application

Start the application using Node.js:

node ~/app.js

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject : Advanced Cloud Computing (P)

Name of the Student: **Prakhar Anil Sharma**

PRN: 20220801121

Title of Practical: **AWS Simple Queue Service (SQS) For Sending Messages**

Your application should now be running at <http://your-ec2-public-ip:3000>.



Home Page

Welcome to the home page!

[Go to Send Page](#)
[View Messages](#)



Send Message

Message:

[Go to Home Page](#)



Messages from SQS Queue

• Hello

[Go to Home Page](#)