D Y PATIL INTERNATIONAL UNIVERSITY
AKURDI PUNE

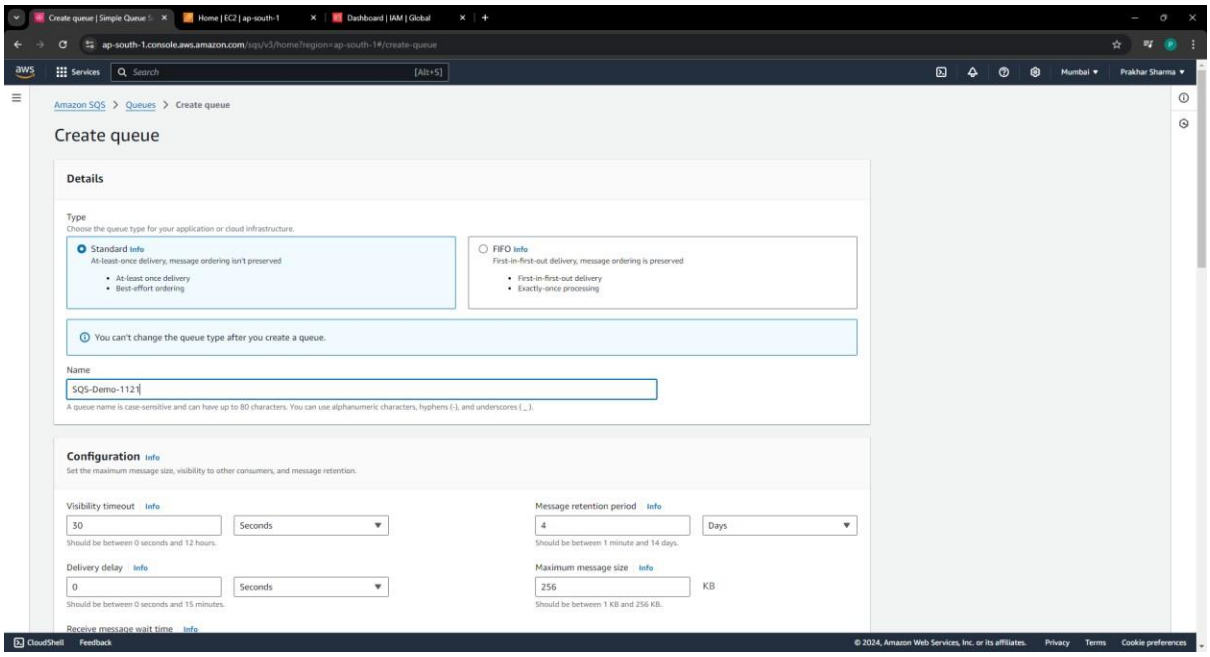
School of Computer Science, Engineering and Applications(SCSEA)
B.C.A. TY (CCSA)
Subject : Advanced Cloud Computing (P)

Name of the Student: **Prakhar Anil Sharma** **PRN: 20220801121**

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

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.



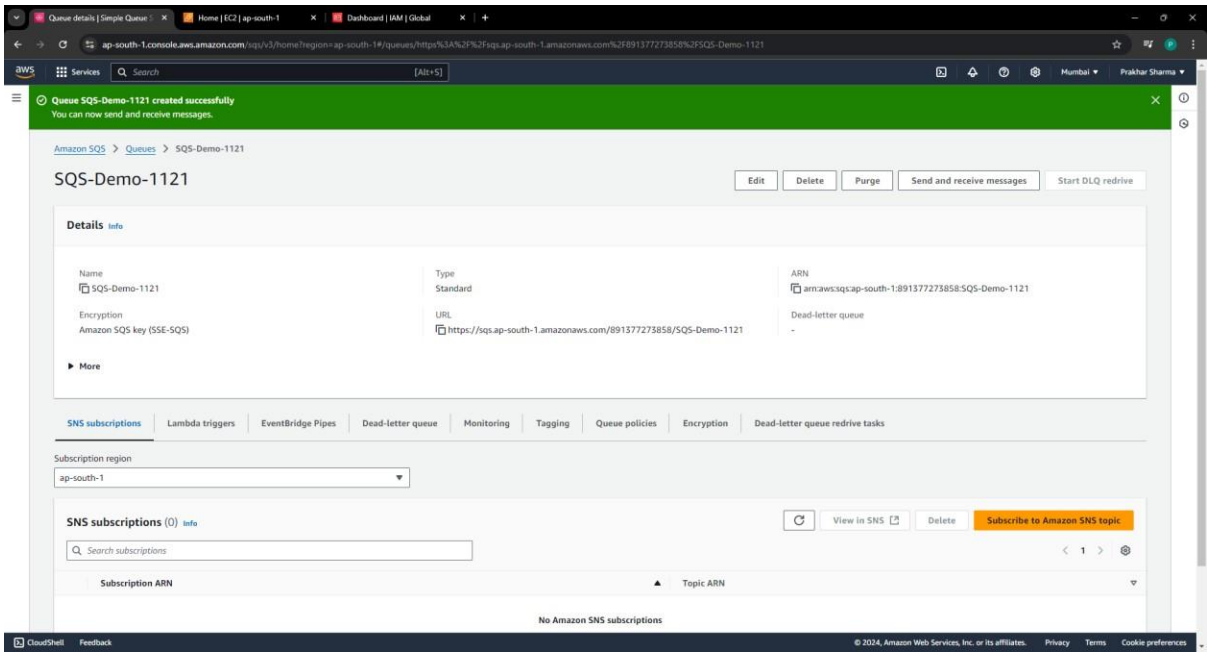
4. Scroll down and click **Create Queue**.

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)

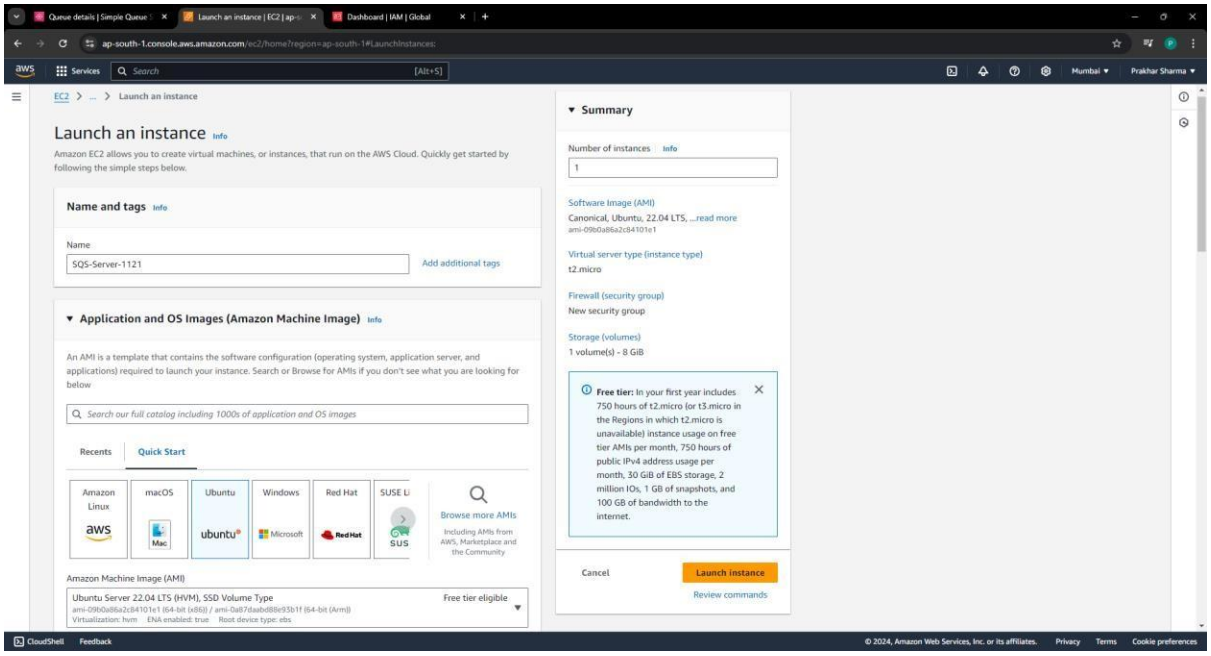
Name of the Student: **Prakhar Anil Sharma** **PRN: 20220801121**

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

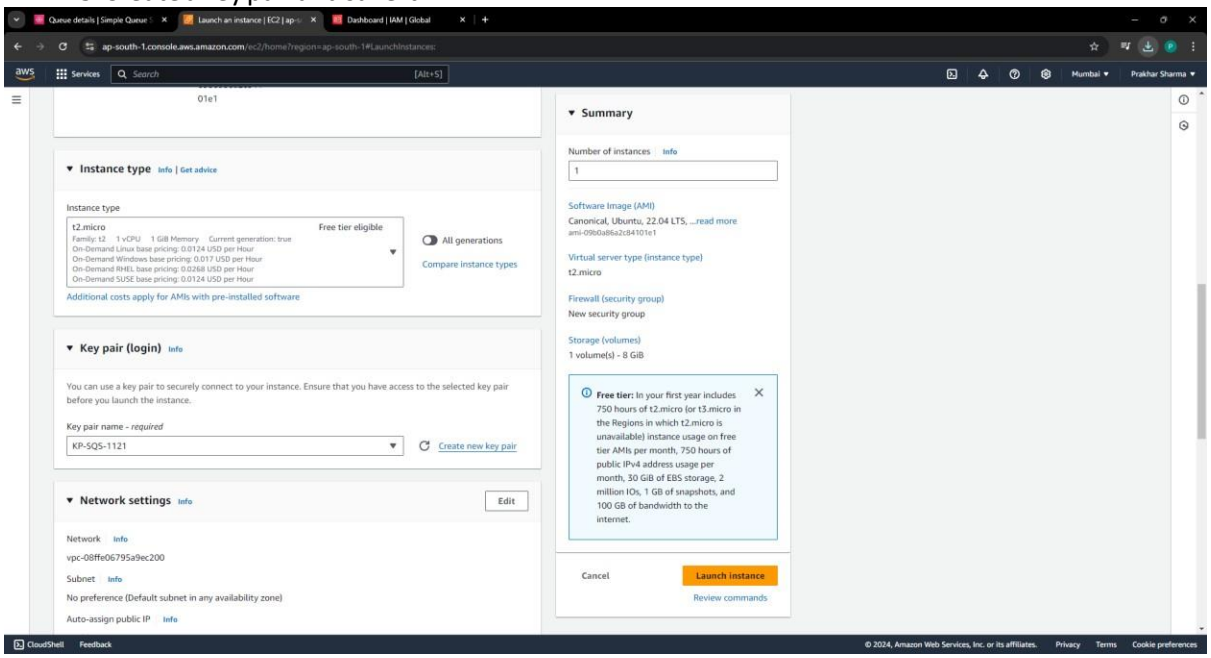


1. Launch an EC2 Instance

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.

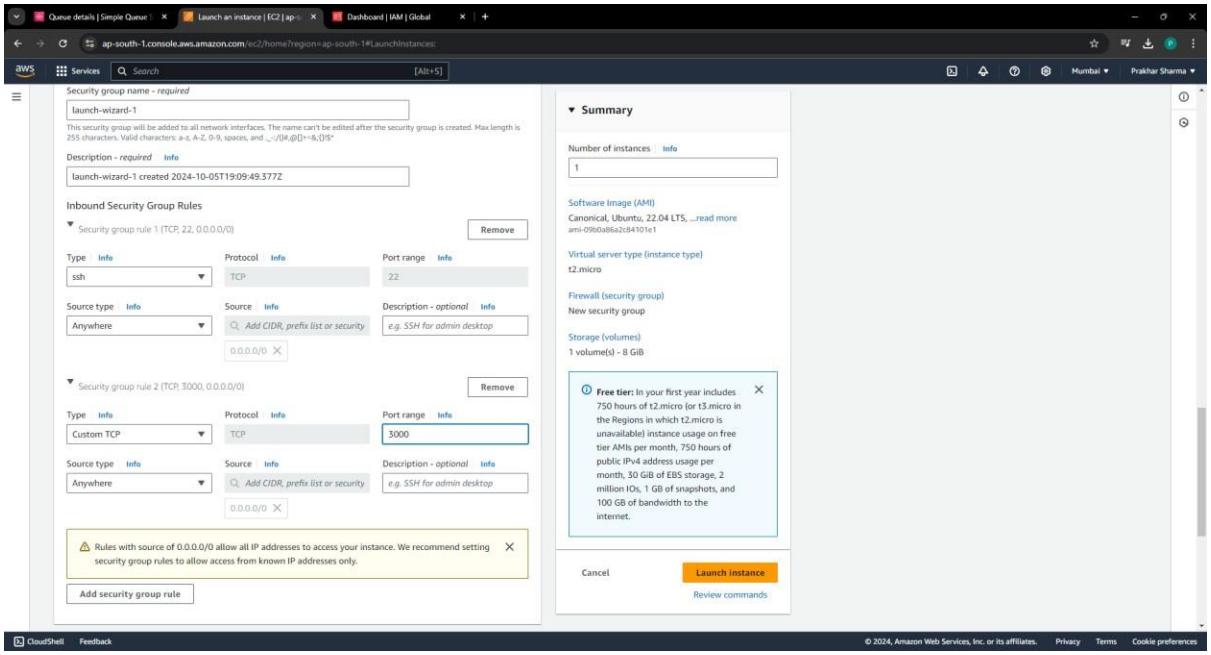


3.Create a key pair and save it.

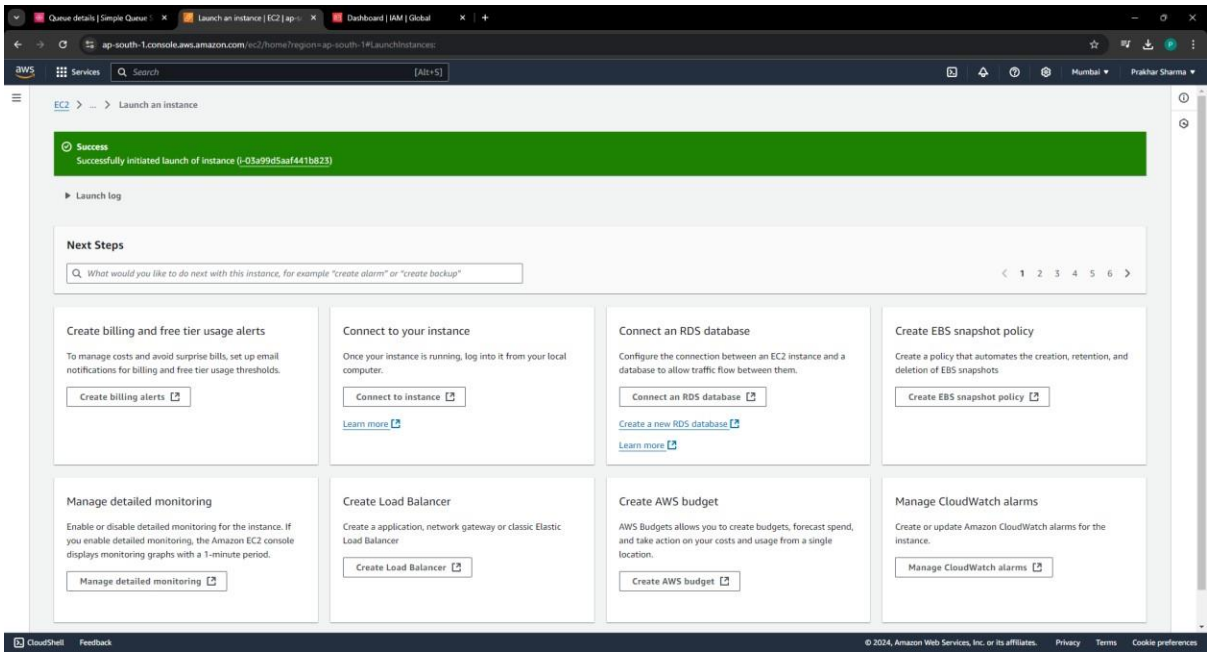


4.In the Network section, click on **Edit** and select **Auto-assign IP**. Enable it and click **Add**.

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

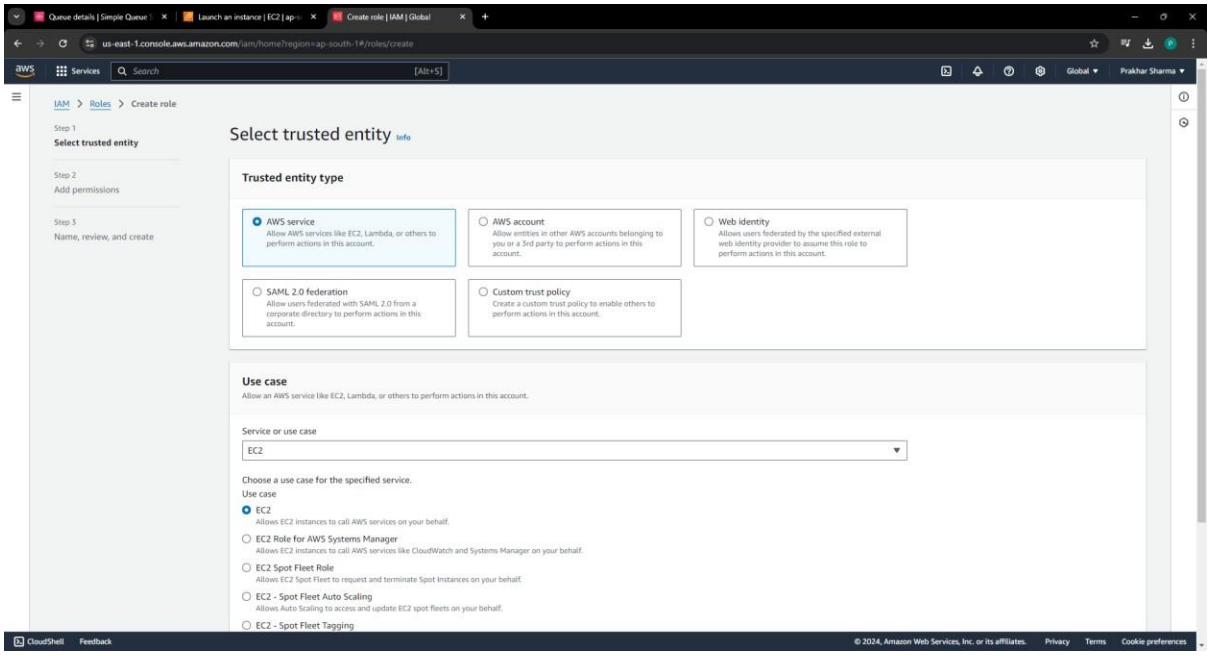


6.Click on **Launch Instance**.

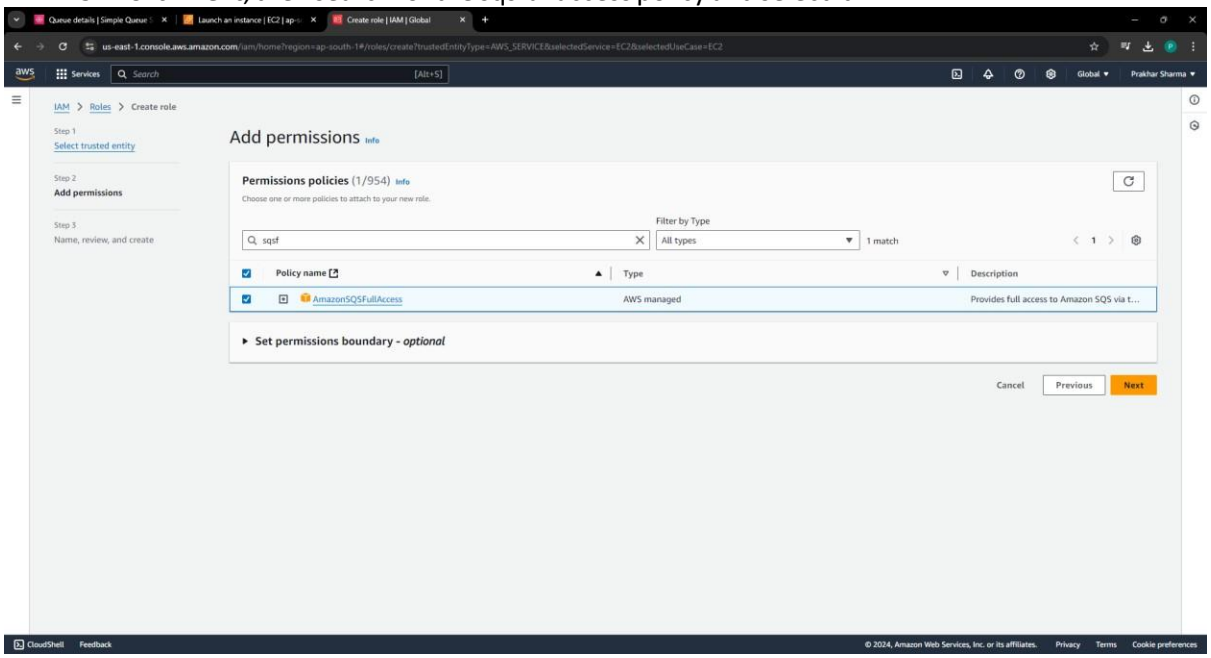


2. Create an IAM Role

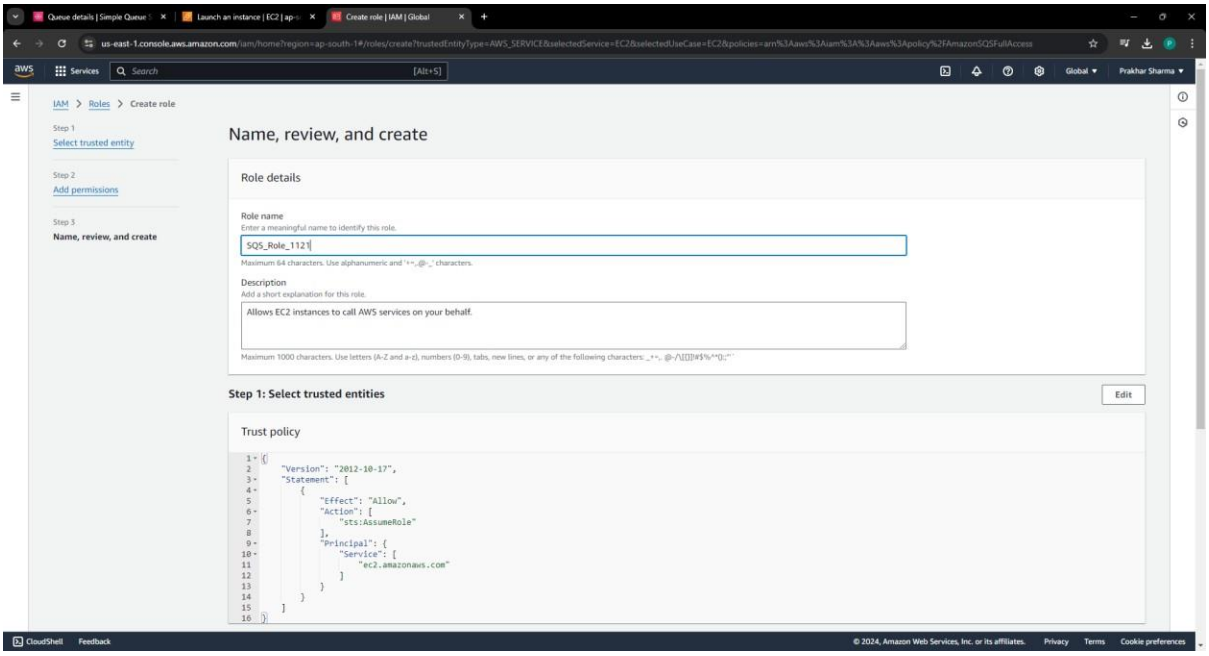
1. Search for IAM (Identity Access Management).
2. In IAM, on the left-side panel, click on **Roles**.
3. Click on **Create Role**.
4. In the **Select trusted entity** section, choose **AWS Service** and select **EC2** in the use case.



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



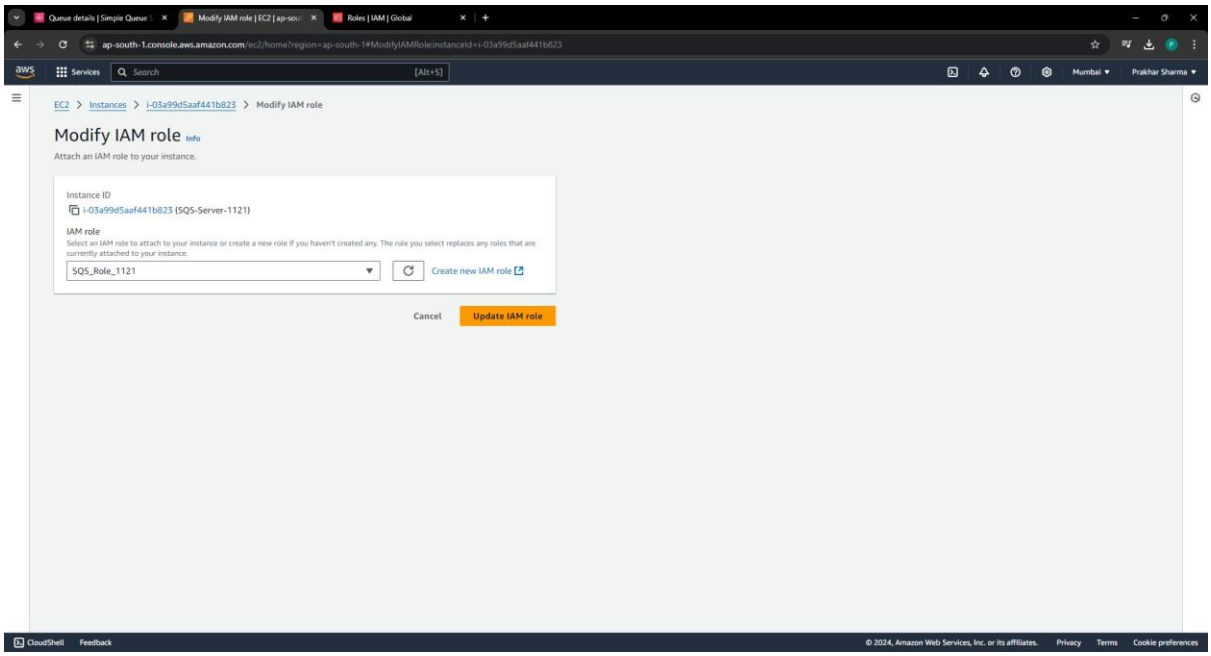
6. Click **Next**, give a name to the role, and click **Create Role**.



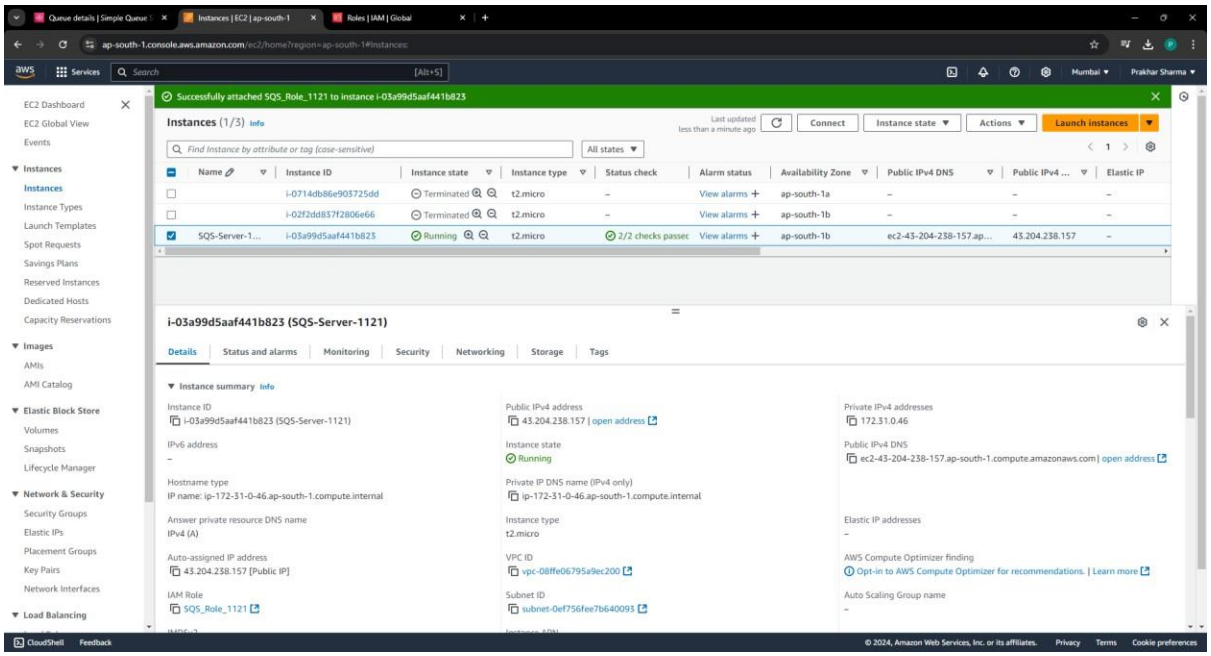
7. After creating the role successfully, navigate back to your previously created EC2instance.

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

9. Select the role you just created.

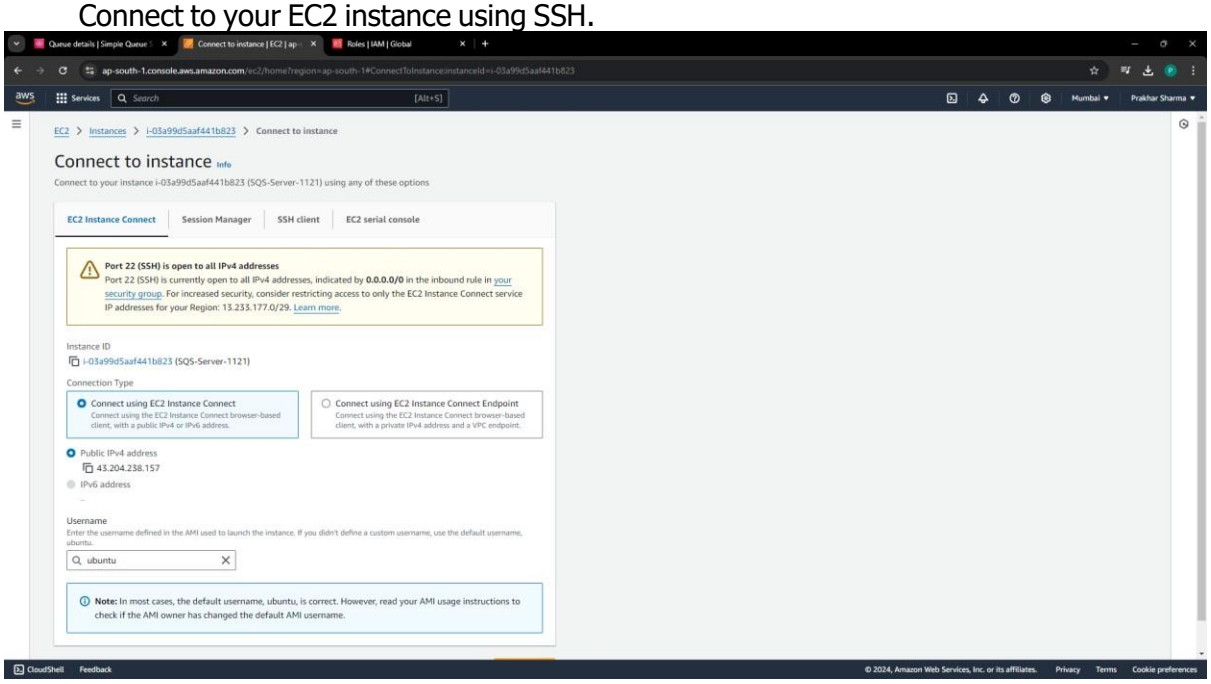


10.Click **Update IAM Role**.



4. Connect to the EC2 Instance

Connect to your EC2 instance using SSH.

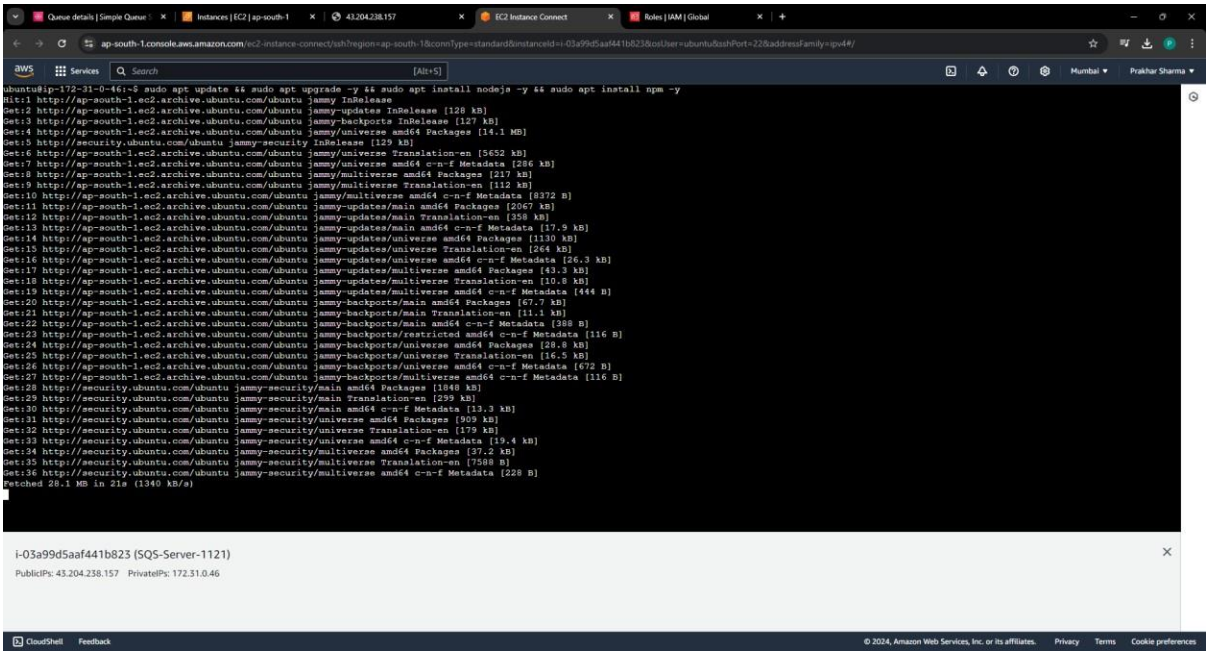


5. Update and Set Up the EC2 Instance

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 installnpm -y



5.3Create app.js and Add Content

1]Create the app.js

file:

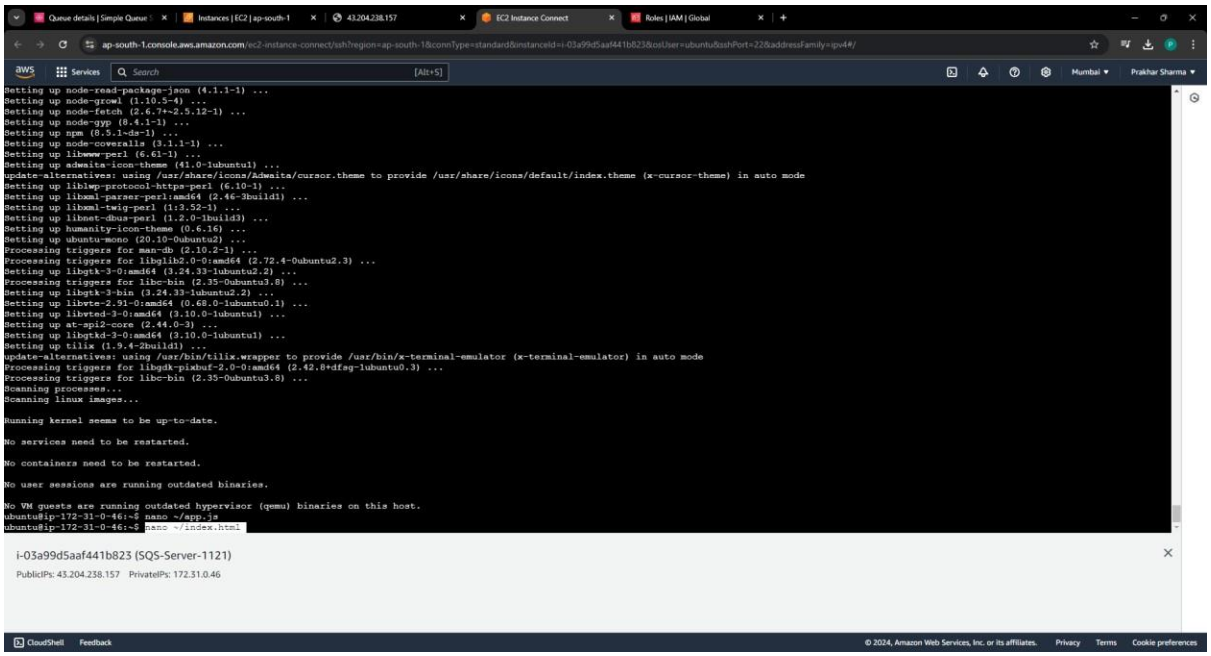
nano ~/app.js

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**



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 =
```

```
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'));
```

```
});

// 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) {
```

```
        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);
```

```
    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}`);
});
```

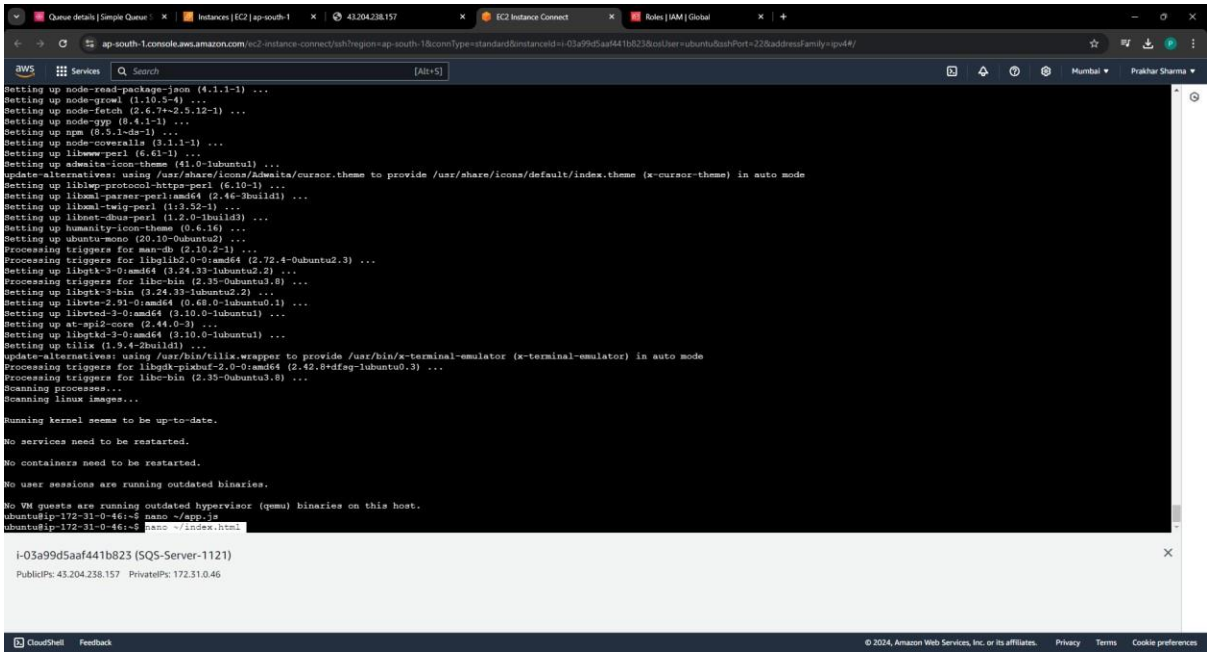
3. Save and close the file:

- 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

1. Create the index.html file:

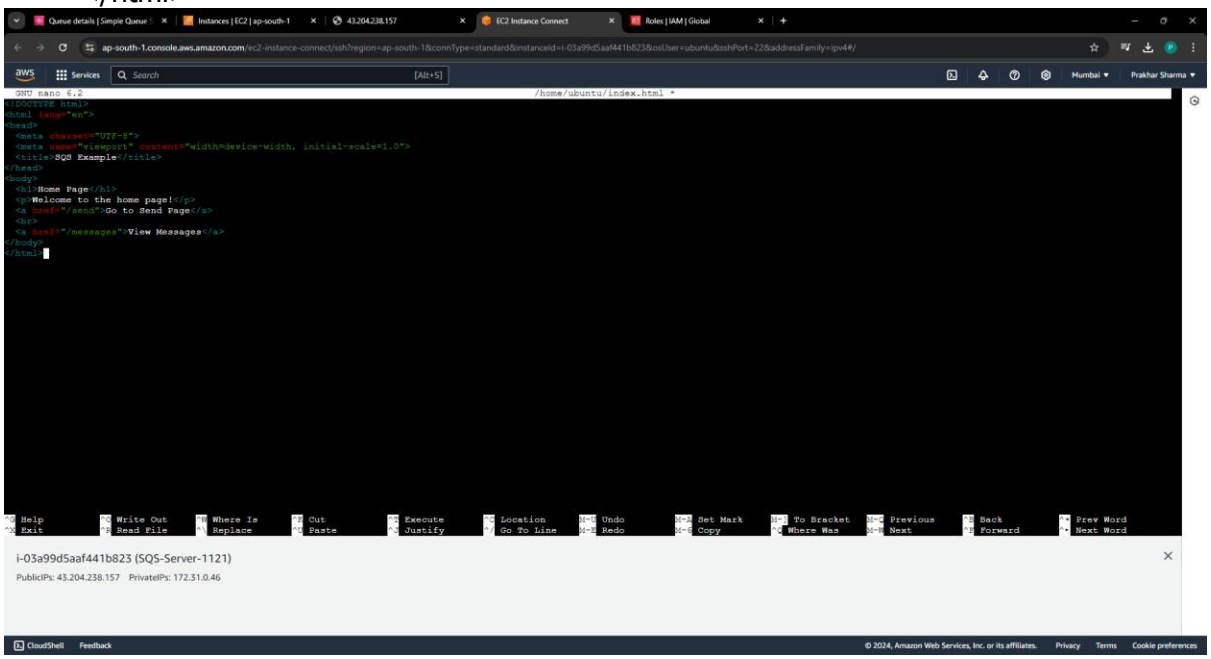
nano ~/index.html



2. Copy and paste the following content into index.html:

```
<!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>Home Page</h1>
```

```
<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:

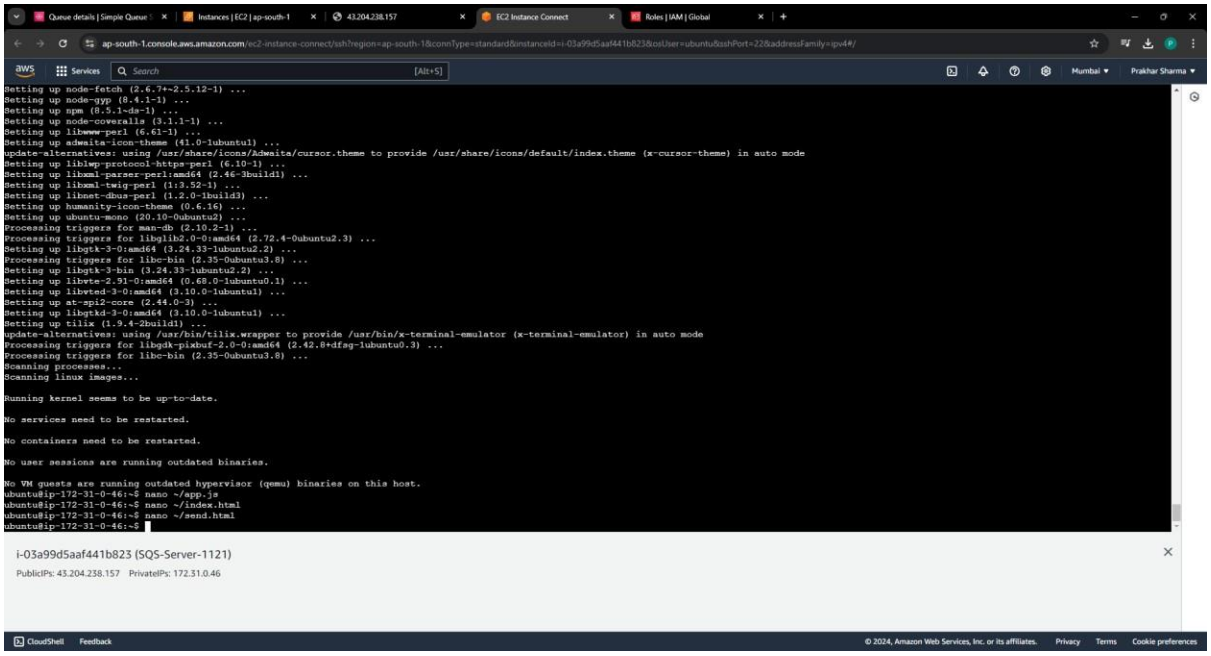
- Press Ctrl + O to save the file.
- Press Enter to confirm.
- Press Ctrl + X to exit the editor.

5.3. Create send.html and Add Content

1. Create the

send.html file:

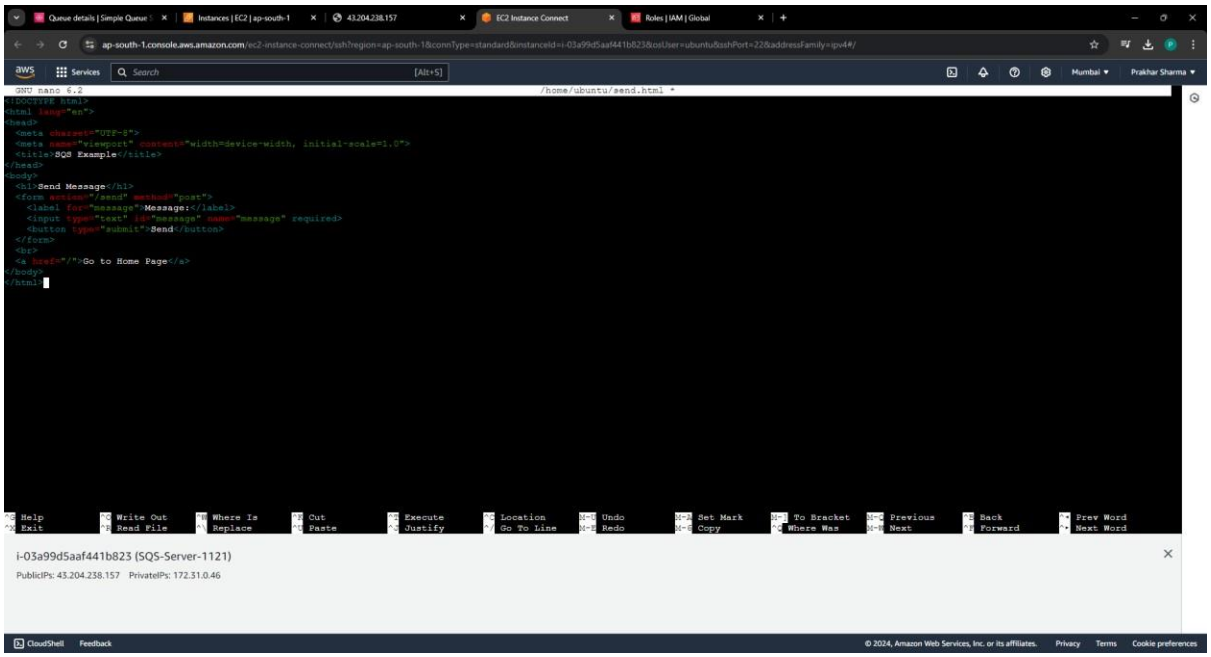
nano ~/send.html



2.Copy and paste the following content into send.html:

```
<!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>Send Message</h1>
```

```
<form action="/send" method="post">
  <label for="message">Message:</label>
  <input type="text" id="message" name="message" required>
  <button  type="submit">Send</button>
</form>
<br>
<a href="/">Go to Home Page</a>
</body>
</html>
```

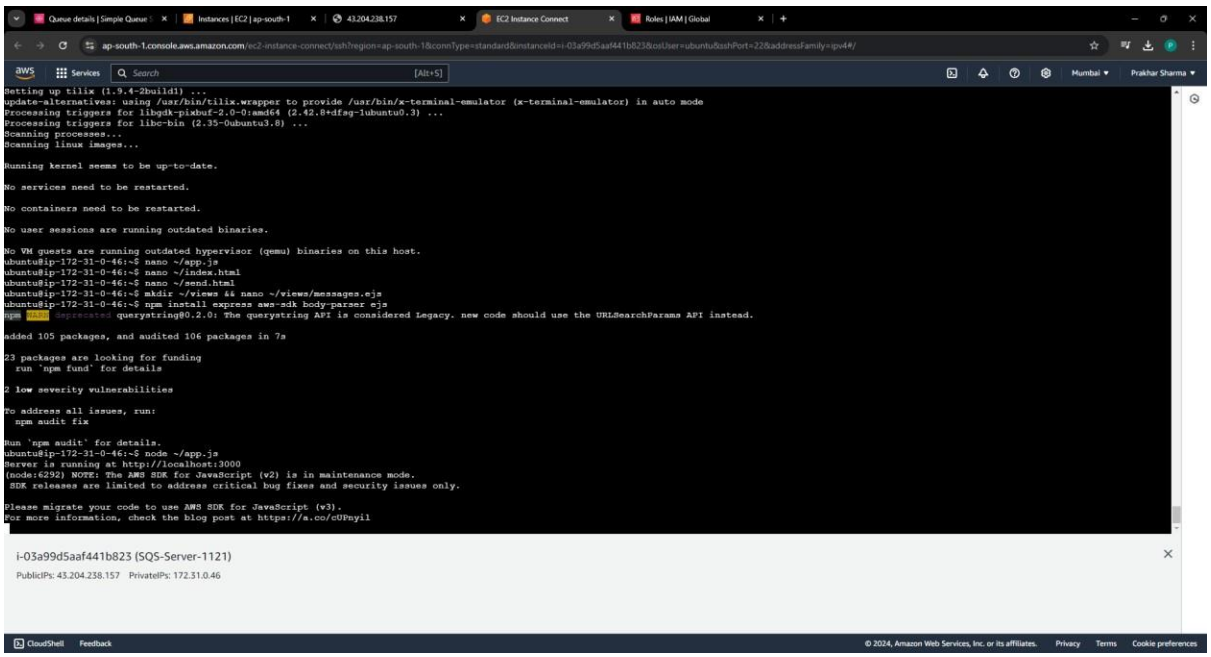


3. Save and close the file:

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

5.4. 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

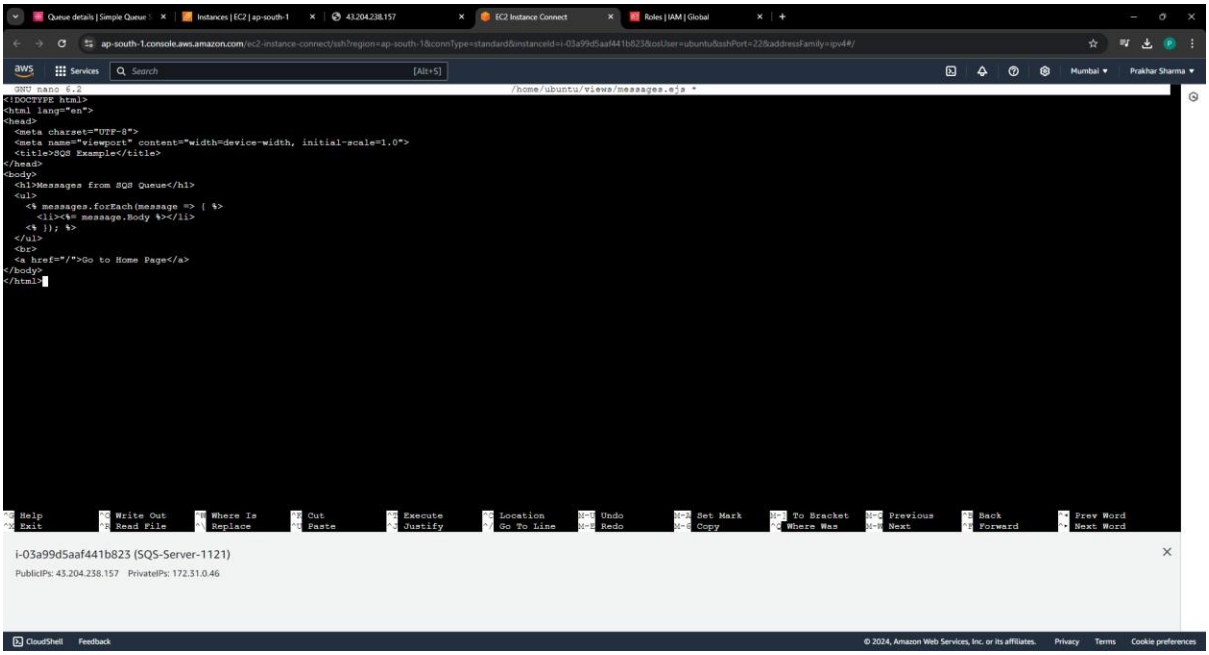


2. Copy and paste the following content into messages.ejs:

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

```
 <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>
```

```
</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

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