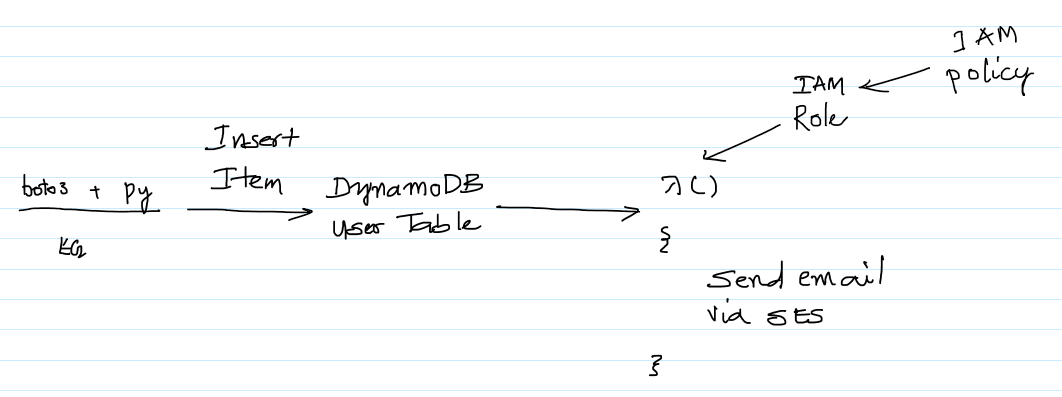
**Use Case**

Let’s take the case of an ecommerce application, users would be creating an account by providing their profile details. In this process the users table is populated with new items in the DynamoDB table (similar to rows in the RDBMS).

When a new user details are populated in the users table, as part of the workflow we would like to get a notification via email or notify some other application. In this use case, once the front-end web application inserts an item into the DynamoDB table, the Lambda function will be automatically called which sends an email for the sake of notification. The functionality in the Lambda can be simply replaced with the code to notify some other application (like marketing application) via SQS.



**AWS Services:** DynamoDB, Lambda, IAM and SES

-- Go to the SES Console, click on “Verify a New Email Address”.

A screenshot of a social media post

Description automatically generated

-- Enter the email address and click on “Verify This Email Address”.

A screenshot of a cell phone

Description automatically generated

-- An email will be sent to this address with a link for the sake of verification. Click on Close.

A screenshot of a cell phone

Description automatically generated

-- Initially the email will be in a “pending verification” status and after clicking on the link in the email, the status of the email will change to “verified”. This is to make sure that the SES service is not used for spamming.

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

-- Follow the same steps and verify another email address. Once of the email would be acting as the sender and the other as the receiver.

-- Now it’s time to create an IAM Role for Lambda. Go to the IAM Management Console, click on Roles and click on “Create role”.

A screenshot of a cell phone

Description automatically generated

-- Select Lambda as the service which is going to use this Role. Click on “Next: Permissions”.

A screenshot of a cell phone

Description automatically generated

-- Select the AWSLambdaDynamoDBExecutionRole and AmazonSESFullAccess policies and click on “Next : Tags”.

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

-- Tags are optional. Simply, click on “Next : Review”.

A screenshot of a social media post

Description automatically generated

-- Give the role a name and click on “Create role”.

A screenshot of a cell phone

Description automatically generated

-- Go to the Lambda Management Console and click on “Create function”.

A screenshot of a cell phone

Description automatically generated

-- Enter the Function name, select the role and NodeJS 10.x or Python2.7 and select the role created earlier. The Python and NodeJS code for sending emails via SES has been mentioned in the next sections. Use either of these languages for the Lambda as per your comfort.

A screenshot of a social media post

Description automatically generated

-- Copy the below NodeJS code. Make sure to replace the from and the to address with the email address which have been verified earlier. And finally click on Save.

*var aws = require('aws-sdk');*

*var ses = new aws.SES({region: 'us-east-1'});*

*exports.handler = function(event, context) {*

*console.log("Incoming: ", event);*

*// var output = querystring.parse(event);*

*var eParams = {*

*Destination: {*

*ToAddresses: ["abc@gmail.com"]//give the email ID which is verified by SES*

*},*

*Message: {*

*Body: {Text: {*

*Data: "Hurray a new user has been created !!!!"*

*}*

*},*

*Subject: { Data: "New User"}*

*},*

*Source: "xyz@gmail.com" //give the email ID which is verified by SES*

*};*

*console.log('===SENDING EMAIL===');*

*var email = ses.sendEmail(eParams, function(err, data){*

*if(err) console.log(err);*

*else {*

*console.log("===EMAIL SENT===");*

*console.log(data);*

*console.log("EMAIL CODE END");*

*console.log('EMAIL: ', email);*

*context.succeed(event);*

*}*

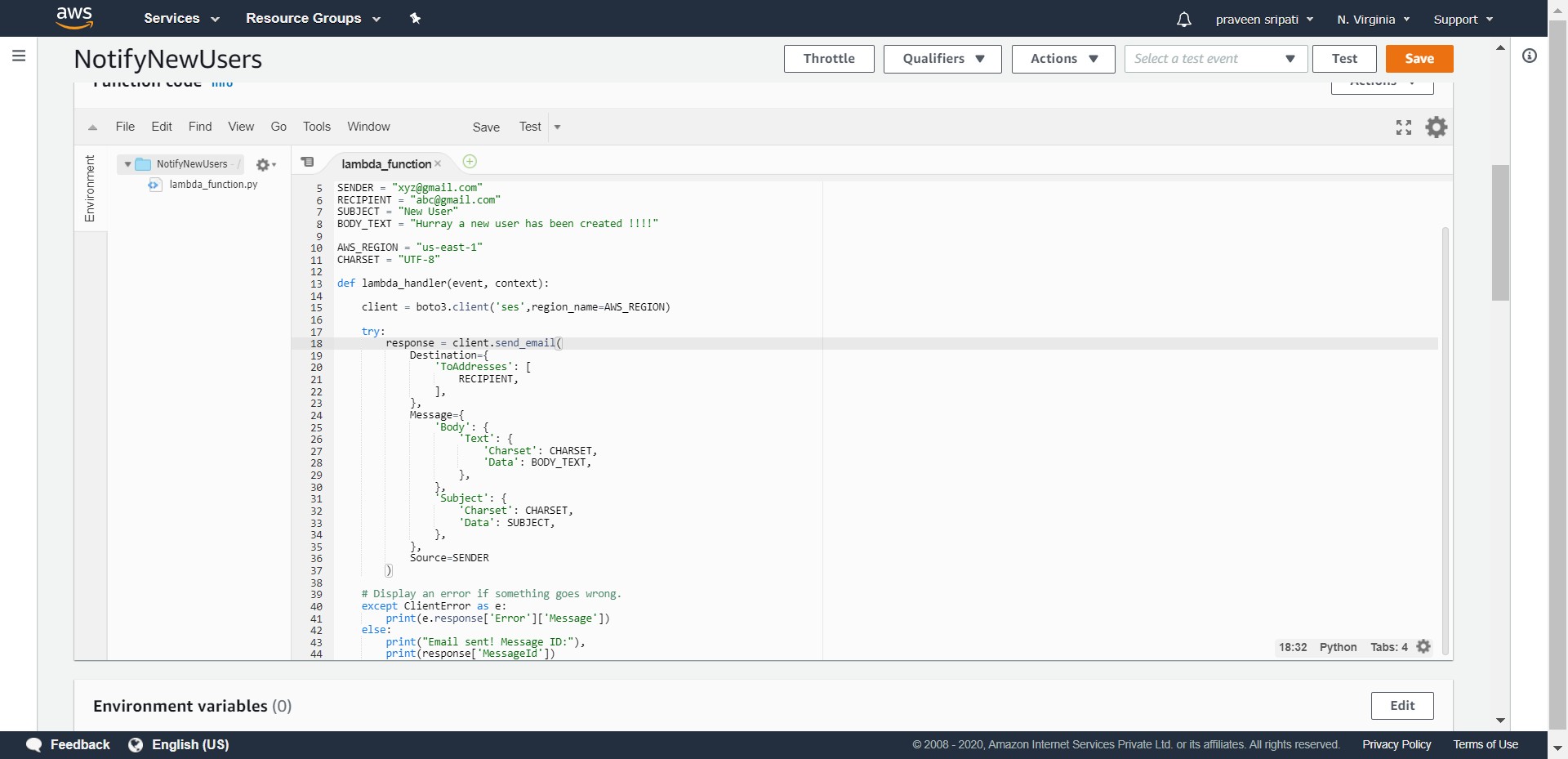
*});*

*};*

A screenshot of a cell phone

Description automatically generated

-- Copy the below Python code. Make sure to replace the from and the to address with the email address which have been verified earlier. And finally click on Save.  
  
import boto3  
from botocore.exceptions import ClientError

print('Loading function')  
  
SENDER = "[xyz@gmail.com](mailto:xyz@gmail.com)"  
RECIPIENT = "[abc@gmail.com](mailto:abc@gmail.com)"  
SUBJECT = "New User"  
BODY\_TEXT = "Hurray a new user has been created !!!!"  
  
AWS\_REGION = "us-east-1"  
CHARSET = "UTF-8"  
  
def lambda\_handler(event, context):  
  
 client = boto3.client('ses',region\_name=AWS\_REGION)  
  
 try:  
 response = client.send\_email(  
 Destination={  
 'ToAddresses': [  
 RECIPIENT,  
 ],  
 },  
 Message={  
 'Body': {  
 'Text': {  
 'Charset': CHARSET,  
 'Data': BODY\_TEXT,  
 },  
 },  
 'Subject': {  
 'Charset': CHARSET,  
 'Data': SUBJECT,  
 },  
 },  
 Source=SENDER  
 )  
  
 # Error handling  
 except ClientError as e:  
 print(e.response['Error']['Message'])  
 else:  
 print("Email sent! Message ID:"),  
 print(response['MessageId'])  
  


-- Go to the DynamoDB Management Console and click on Create table. Enter the below details and click on Create. The table would be created in a few seconds.

*Table name – users*

*Primary key – userid (Number)*

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

-- Click on “Manage Streams”.

A screenshot of a social media post

Description automatically generated

-- Go with the default options and click on “Enable”.

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

-- Go back to the Lambda Management Console and click on “Add trigger”.

A screenshot of a social media post

Description automatically generated

-- Select the DynamoDB service and select the “users” table created earlier. Go with the rest of the default options and click on “Add”. This integrates the DynamoDB table with the Lambda function via Triggers.

A screenshot of a cell phone

Description automatically generated

-- The DynamoDB Trigger should be added to the Lambda function as shown below.

A screenshot of a social media post

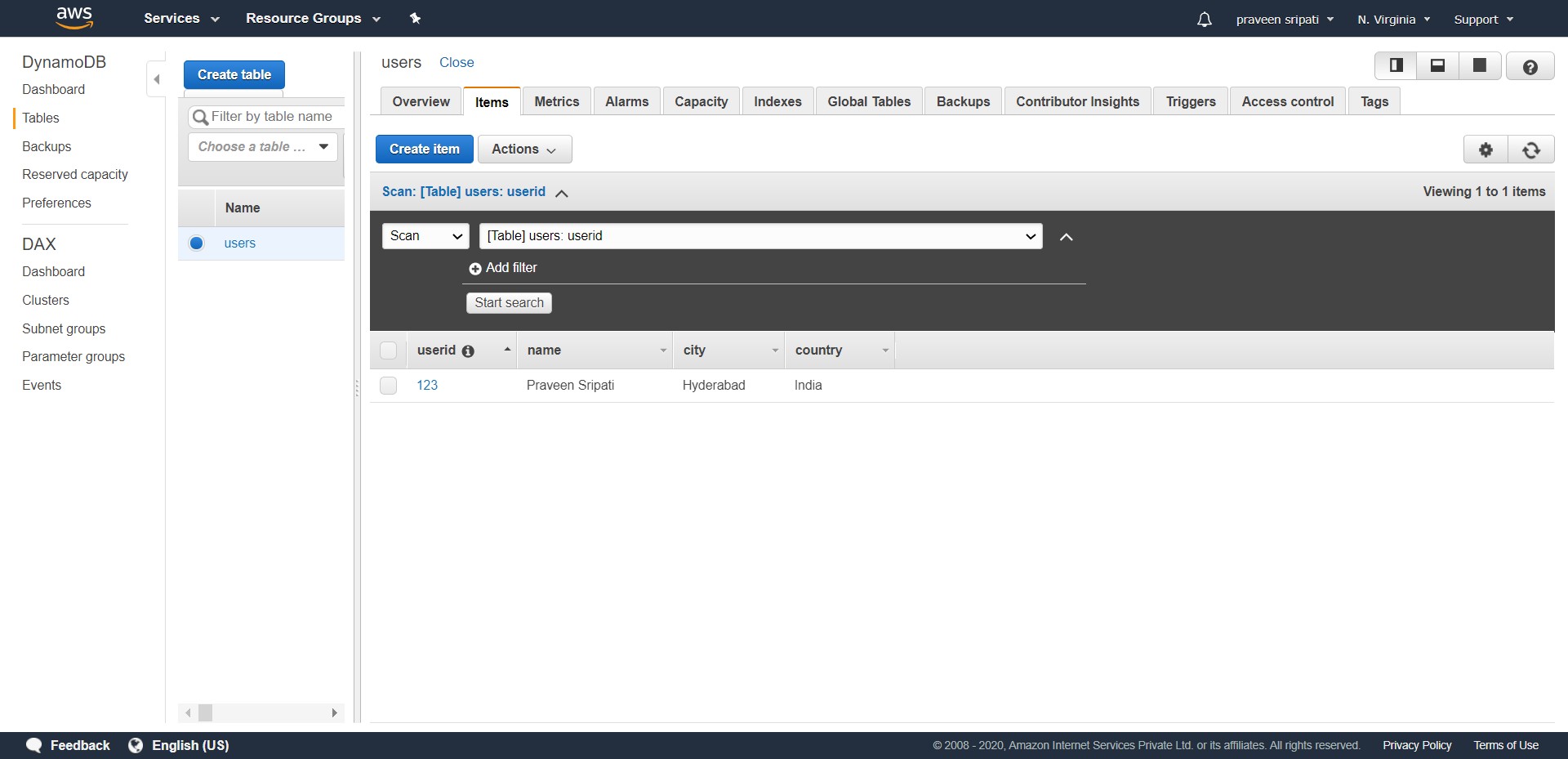
Description automatically generated

-- Create an IAM Role for EC2 with the AmazonDynamoDBFullAccess Policy attached to it.

-- Create an EC2 instance with the below details and connect to it.  
  
 -- t2.micro  
 -- Ubuntu OS  
 -- Security Group with Port 22/Inbound allowed  
  
-- Once connected to the EC2 instance execute the below commands. These commands will install Python, pip, boto3 (AWS Python SDK).  
  
#become root  
sudo su  
  
#get the list of softwares  
apt-get update  
  
#install python and pip  
apt-get install python2.7 python-pip -y  
  
#install python aws sdk  
pip install boto3  
  
exit  
mkdir .aws  
echo -e "[default]\nregion=us-east-1" > .aws/config

-- Create a file called dynamodb-put.py with the below content. Finally execute the “python dynamodb-put.py” to execute the python program which inserts an item in the DynamoDB table. Here we are trying to mimic an application inserting an item in the DynamoDB table.  
  
import boto3  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 dynamodb = boto3.resource('dynamodb')  
 table = dynamodb.Table('users')  
  
 response = table.put\_item(  
 Item={  
 'userid': 123,  
 'name': "Praveen Sripati",  
 'city': "Hyderabad",  
 'country': "India"  
 }  
 )  
 print("Put user succeeded:")

-- The item should be inserted into the DynamoDB table as shown below. This should automatically trigger the Lambda function, which will send an email via AWS SES service.



-- Check your email and there should be email from the SES service which has been triggered by the Lambda function. The function in the Lambda function can be replaced with any code for integration with other applications via AWS SQS Service.

A screenshot of a social media post

Description automatically generated