

Step 1: Lambda function creation

Create function [Info](#)

AWS Serverless Application Repository applications have moved to [Create application](#).

☒ **Author from scratch**
Start with a simple Hello World example.

☐ **Use a blueprint**
Build a Lambda application from sample code and configuration presets for common use cases.

☐ **Container image**
Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Architecture [Info](#)
Choose the instruction set architecture you want for your function code.
☒ x86_64
☐ arm64

Permissions [Info](#)
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

☐ Create a new role with basic Lambda permissions

☒ Use an existing role

☐ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

[View the Lambda-Load-Inventory-Role role](#) on the IAM console.

▼ Advanced settings

☐ **Enable Code signing** [Info](#)
Use code signing configurations to ensure that the code has been signed by an approved source and has not been altered since signing.

☐ **Enable function URL** [Info](#)
Use function URLs to assign HTTPS endpoints to your Lambda function.

☐ **Enable tags** [Info](#)
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources, track your AWS costs, and enforce attribute-based access control.

☐ **Enable VPC** [Info](#)
Connect your function to a VPC to access private resources during invocation.

Step 2: Copy paste the code provided into the code environment

The image displays two screenshots of the AWS Lambda console, showing the 'Load-Inventory' function code and its properties.

Top Screenshot: The 'Code source' tab is selected, showing the Python code for the 'Load-Inventory' function. The code is a Lambda function that triggers on an S3 event, downloads a file from the bucket, and inserts it into a DynamoDB table. The code is as follows:

```
1 # Load-Inventory Lambda Function
2 #
3 # This function is triggered by an object being created in an Amazon S3 bucket.
4 # The file is downloaded and each line is inserted into a DynamoDB table.
5 import json, urllib, boto3, csv
6 # Connect to S3 and Dynamo
7 s3 = boto3.resource('s3')
8 dynamodb = boto3.resource('dynamodb')
9 # Connect to the DynamoDB tables
10 inventoryTable = dynamodb.Table('Inventory')
11 # This handler is run every time the Lambda function is triggered
12 def lambda_handler(event, context):
13     # Show the incoming event in the debug log
14     print('Event received by Lambda function: ' + json.dumps(event, indent=2))
15     # Get the bucket and object key from the event
16     bucket = event['Records'][0]['s3']['bucket']['name']
17     key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'])
18     localFilename = '/tmp/inventory.txt'
19     # Download the file from S3 to the local filesystem
20     try:
21         s3.meta.client.download_file(bucket, key, localFilename)
22     except Exception as e:
23         print('Error getting object {} from bucket {}. Make sure they exist and your bucket is in the same region as this function.'.format(key, bucket))
24         raise e
25     # Read the Inventory CSV file
26     with open(localFilename) as csvfile:
27         reader = csv.DictReader(csvfile, delimiter=',')
28         # Read each row in the file
29         rowCount = 0
30         for row in reader:
31             rowCount += 1
32             # Show the row in the debug log
33             print(row['store'], row['item'], row['count'])
34             try:
35                 # Insert Store, Item and Count into the Inventory table
36                 inventoryTable.put_item(
```

Code properties:

- Package size: 299.0 byte
- SHA256 hash: HAPq9ERuJVECSgLvncJgyd5vZtd9uU6F932tQbXy=
- Last modified: November 4, 2023 at 04:57 PM GMT+7

Bottom Screenshot: The 'Code source' tab is selected, showing the same Python code for the 'Load-Inventory' function. The code is as follows:

```
1 # Load-Inventory Lambda Function
2 #
3 # This function is triggered by an object being created in an Amazon S3 bucket.
4 # The file is downloaded and each line is inserted into a DynamoDB table.
5 import json, urllib, boto3, csv
6 # Connect to S3 and Dynamo
7 s3 = boto3.resource('s3')
8 dynamodb = boto3.resource('dynamodb')
9 # Connect to the DynamoDB tables
10 inventoryTable = dynamodb.Table('Inventory')
11 # This handler is run every time the Lambda function is triggered
12 def lambda_handler(event, context):
13     # Show the incoming event in the debug log
14     print('Event received by Lambda function: ' + json.dumps(event, indent=2))
15     # Get the bucket and object key from the event
16     bucket = event['Records'][0]['s3']['bucket']['name']
17     key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'])
18     localFilename = '/tmp/inventory.txt'
19     # Download the file from S3 to the local filesystem
20     try:
21         s3.meta.client.download_file(bucket, key, localFilename)
22     except Exception as e:
23         print('Error getting object {} from bucket {}. Make sure they exist and your bucket is in the same region as this function.'.format(key, bucket))
24         raise e
25     # Read the Inventory CSV file
26     with open(localFilename) as csvfile:
27         reader = csv.DictReader(csvfile, delimiter=',')
28         # Read each row in the file
29         rowCount = 0
30         for row in reader:
31             rowCount += 1
32             # Show the row in the debug log
33             print(row['store'], row['item'], row['count'])
34             try:
35                 # Insert Store, Item and Count into the Inventory table
36                 inventoryTable.put_item(
```

Code properties:

- Package size: 989.0 byte
- SHA256 hash: 7EH6F6pY173bp5nJ+OWE9nd+TZ4D/PNAPMLYmJ2yA=
- Last modified: November 4, 2023 at 04:57 PM GMT+7

Step 2: Create S3 bucket

The screenshot shows the 'Create bucket' page in the AWS Management Console. The browser address bar shows the URL: `s3.console.aws.amazon.com/s3/bucket/create?region=us-east-1`. The page title is 'Create bucket' with an 'Info' link. Below the title, it says 'Buckets are containers for data stored in S3. [Learn more](#)'. The 'General configuration' section contains a 'Bucket name' field with 'inventory-12345' and a note that bucket names must be unique. The 'AWS Region' is set to 'US East (N. Virginia) us-east-1'. There is a 'Copy settings from existing bucket - optional' section with a 'Choose bucket' button. The 'Object Ownership' section has two radio buttons: 'ACLs disabled (recommended)' (selected) and 'ACLs enabled'. The 'Block Public Access settings for this bucket' section has a note about public access. The bottom of the page shows the Windows taskbar with the date 4/11/2023 and time 1:02 PM.

Create bucket [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name
inventory-12345
Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

AWS Region
US East (N. Virginia) us-east-1

Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.
Choose bucket

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☒ ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☐ ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership
Bucket owner enforced

Block Public Access settings for this bucket
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to

2.2: create eventnotification

The screenshot shows the 'Create event notification' page in the AWS Management Console. The browser address bar shows the URL: `s3.console.aws.amazon.com/s3/bucket/inventory-12345/property/notification/create?region=us-east-1`. The page title is 'Create event notification' with an 'Info' link. Below the title, it says 'To enable notifications, you must first add a notification configuration that identifies the events you want Amazon S3 to publish and the destinations where you want Amazon S3 to send the notifications.' The 'General configuration' section contains an 'Event name' field with 'Load-Inventory', a 'Prefix - optional' field with 'images/', and a 'Suffix - optional' field with 'jpg'. The 'Event types' section has a note about specifying at least one event. The 'Object creation' section has three checkboxes: 'All object create events' (checked), 'Put', and 'Post'. The bottom of the page shows the Windows taskbar with the date 4/11/2023 and time 1:04 PM.

Create event notification [Info](#)

To enable notifications, you must first add a notification configuration that identifies the events you want Amazon S3 to publish and the destinations where you want Amazon S3 to send the notifications.

General configuration

Event name
Load-Inventory
Event name can contain up to 255 characters.

Prefix - optional
Limit the notifications to objects with key starting with specified characters.
images/

Suffix - optional
Limit the notifications to objects with key ending with specified characters.
jpg

Event types
Specify at least one event for which you want to receive notifications. For each group, you can choose an event type for all events, or you can choose one or more individual events.

Object creation

☒ All object create events
s3:ObjectCreated*

☐ Put
s3:ObjectCreated:Put

☐ Post
s3:ObjectCreated:Post

☐ Copy
s3:ObjectCreated:Copy

The screenshot displays the AWS Management Console interface for an Amazon S3 bucket named 'inventory-12345'. The browser address bar shows the URL 's3.console.aws.amazon.com/s3/buckets/inventory-12345?region=us-east-1&tab=properties'. A green notification banner at the top states 'Successfully created event notification "Load-Inventory". Operation successfully completed.' The console shows the bucket's overview, including its AWS Region (US East (N. Virginia) us-east-1), Amazon Resource Name (ARN) (arn:aws:s3::inventory-12345), and Creation date (November 4, 2023, 17:02:31 (UTC+07:00)). The 'Bucket Versioning' section indicates that versioning is disabled, and the 'Multi-factor authentication (MFA) delete' is also disabled. The 'Tags' section shows no tags are currently attached to the bucket. The bottom of the screen shows the Windows taskbar with various application icons and the system clock indicating 5:05 PM on 4/11/2023.

Assignment 2023-H109-C02: Module 13 Guided Lab - Implementing Amazon S3 bucket inventory

→ → → s3.console.aws.amazon.com/s3/buckets/inventory-12345?region=us-east-1&tab=properties

BWS Services Search [Alt+S]

Global voclabs/user2753113-10421942@student.wain.edu.au 2633-4025

Successfully created event notification "Load-Inventory".
Operation successfully completed.

Amazon S3 > Buckets > inventory-12345

inventory-12345

Objects Properties Permissions Metrics Management Access Points

Bucket overview

AWS Region US East (N. Virginia) us-east-1	Amazon Resource Name (ARN) arn:aws:s3::inventory-12345	Creation date November 4, 2023, 17:02:31 (UTC+07:00)
---	---	---

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Versioning Disabled

Multi-factor authentication (MFA) delete Disabled

An additional layer of security that requires multi-factor authentication for changing Bucket Versioning settings and permanently deleting object versions. To modify MFA delete settings, use the AWS CLI, AWS SDK, or the Amazon S3 REST API. [Learn more](#)

Tags (0)

You can use bucket tags to track storage costs and organize buckets. [Learn more](#)

Key	Value
-----	-------

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

5:05 PM 4/11/2023

2.3: tetsing

Uploading the inventory file to S3

The screenshot displays the AWS S3 console interface for uploading files to a bucket named 'inventory-12345'. The 'Upload' page shows a list of files to be uploaded, including 'inventory-springfield.txt', 'inventory-shanghai.txt', 'inventory-pusan.txt', 'inventory-karachi.txt', 'inventory-calcutta.txt', and 'inventory-berlin.txt'. The destination is set to 's3://inventory-12345'.

Files and folders (6 Total, 888.0 B)

Name	Folder	Type	Size
inventory-springfield.txt	-	text/plain	168.0 B
inventory-shanghai.txt	-	text/plain	152.0 B
inventory-pusan.txt	-	text/plain	133.0 B
inventory-karachi.txt	-	text/plain	145.0 B
inventory-calcutta.txt	-	text/plain	150.0 B
inventory-berlin.txt	-	text/plain	140.0 B

Destination
s3://inventory-12345

Destination details
Bucket settings that impact new objects stored in the specified destination.

Upload: status

The information below will no longer be available after you navigate away from this page.

Summary

Destination	Succeeded	Failed
s3://inventory-12345	6 Files, 888.0 B (100.00%)	0 Files, 0 B (0%)

Files and folders

Files and folders (6 Total, 888.0 B)

Name	Folder	Type	Size	Status	Error
inventory-springfield.txt	-	text/plain	168.0 B	Succeeded	-
inventory-shanghai.txt	-	text/plain	152.0 B	Succeeded	-
inventory-pusan.txt	-	text/plain	133.0 B	Succeeded	-
inventory-karachi.txt	-	text/plain	145.0 B	Succeeded	-
inventory-calcutta.txt	-	text/plain	150.0 B	Succeeded	-
inventory-berlin.txt	-	text/plain	140.0 B	Succeeded	-

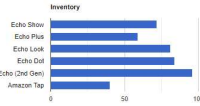
Assignments 2023-HX09-CDS Module 13 Guided Lab - Impl... Upload objects - S3 bucket inv... Inventory System

aws-tc-largeobjects3-us-west-2.amazonaws.com/ILT-TF-200-ACACAD-20-EN/mod13-guided/web/inventory.htm?region=us-east-1&poolid=us-east-1:bba6bf51-90df-4172-b82c-21607ce15283

Inventory Dashboard

Choose a store to view current inventory levels.

Store:	Store	Item	Count
All Stores	All Stores	Echo Show	72
All Stores	All Stores	Echo Plus	59
All Stores	All Stores	Echo Look	81
All Stores	All Stores	Echo Dot	84
All Stores	All Stores	Echo (2nd Gen)	96
All Stores	All Stores	Amazon Tap	40



This page uses an Amazon Cognito identity to retrieve data directly from Amazon DynamoDB.

Type here to search

Assignments 2023-HX09-CDS Module 13 Guided Lab - Impl... Upload objects - S3 bucket inv... View table | Amazon DynamoDB Inventory System

us-east-1.console.aws.amazon.com/dynamodbv2/home?region=us-east-1#table=Inventory

Any tap key
Show key values
Find values by table name

Inventory

Get live item count

When you choose "Start scan," you will perform a DynamoDB scan to determine the most-recent item count. This scan might consume additional table read capacity units.

It is not recommended to perform this action on very large tables or tables that serve critical production traffic. You can pause the action at any time to avoid consuming extra read capacity.

Item count: 36
Scan status: Complete
Last updated: November 04, 2023 17:13:10

Scan again

Cancel

Table status: Active

Get live item count

Table capacity metrics

Read usage (average units/second)
Read throttled requests (count)
Read throttled events (count)

Inventory

▼ Scan or query items

☒ Scan ☐ Query

Select a table or index: **Table - Inventory** | Select attribute projection: **All attributes**

Filters:

Run **Reset**

Completed. Read capacity units consumed: 0.5

Items returned (36)

	Store (String)	Item (String)	Count
<input type="checkbox"/>	Calcutta	Amazon Tap	15
<input type="checkbox"/>	Calcutta	Echo (2nd Gen)	0
<input type="checkbox"/>	Calcutta	Echo Dot	7
<input type="checkbox"/>	Calcutta	Echo Look	3
<input type="checkbox"/>	Calcutta	Echo Plus	16

Step 3: Create Amazon SNS

Create topic

Details

Type: **Standard**

- Best-effort message ordering
- At-least once message delivery
- Highest throughput in publishes/second
- Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints

Name: **NoStock**

Display name - optional: **My Topic**

Encryption - optional

Access policy - optional

Assignments 2023-H009-COS2... x Module 13 Guided Lab - Impl... x Upload objects - S3 bucket m... x Items | Amazon DynamoDB Mi... x NoStock | Topics | Amazon SNS x Inventory System x +

us-east-1.console.aws.amazon.com/sns/v3/home?region=us-east-1#/topic/amawsnsus-east-1-263340255179/NoStock

aws Services Search [Alt+S]

N. Virginia voclabs/user2753113+104219428@student.swin.edu.au @ 2633-4025...

Amazon SNS

Dashboard
Topics
Subscriptions
Mobile
Push notifications
Text messaging (SMS)
Origination numbers

New Feature
Amazon SNS now supports in-place message archiving and replay for FIFO topics. [Learn more](#)

Topic NoStock created successfully.
You can create subscriptions and send messages to them from this topic. [Publish message](#)

[Amazon SNS](#) > [Topics](#) > NoStock

NoStock [Edit](#) [Delete](#) [Publish message](#)

Details

Name NoStock	Display name -
ARN amawsnsus-east-1-263340255179:NoStock	Topic owner 263340255179
Type Standard	

[Subscriptions](#) [Access policy](#) [Data protection policy](#) [Delivery policy \(HTTP/S\)](#) [Delivery status logging](#) [Encryption](#) [Tags](#) [Integrations](#)

Subscriptions (0) [Edit](#) [Delete](#) [Request confirmation](#) [Confirm subscription](#) [Create subscription](#)

Search

ID	Endpoint	Status	Protocol
No subscriptions found You don't have any subscriptions to this topic.			

[Create subscription](#)

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Assignments 2023-H009-COS2... x Module 13 Guided Lab - Impl... x Upload objects - S3 bucket m... x Items | Amazon DynamoDB Mi... x Create subscription | Subscrip... x Inventory System x +

us-east-1.console.aws.amazon.com/sns/v3/home?region=us-east-1#/create-subscription

aws Services Search [Alt+S]

N. Virginia voclabs/user2753113+104219428@student.swin.edu.au @ 2633-4025...

Amazon SNS

[Amazon SNS](#) > [Subscriptions](#) > Create subscription

Create subscription

Details

Topic ARN
amawsnsus-east-1-263340255179:NoStock

Protocol
The type of endpoint to subscribe
Email

Endpoint
An email address that can receive notifications from Amazon SNS.
binhngwh01067@fpt.edu.vn

After your subscription is created, you must confirm it. [info](#)

Subscription filter policy - optional [info](#)
This policy filters the messages that a subscriber receives.

Redrive policy (dead-letter queue) - optional [info](#)
Send undeliverable messages to a dead-letter queue.

[Cancel](#) [Create subscription](#)

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Amazon SNS

Dashboard
Topics
Subscriptions
Mobile
Push notifications
Text messaging (SMS)
Origination numbers

Subscription to NoStock created successfully.
The ARN of the subscription is `arn:aws:sns:us-east-1:263340255179:NoStock:cd966da2-ba0e-4648-b5a2-7d63c31473bb`.

Amazon SNS > Topics > NoStock > Subscription: cd966da2-ba0e-4648-b5a2-7d63c31473bb

Subscription: cd966da2-ba0e-4648-b5a2-7d63c31473bb

Edit Delete

Details

ARN <code>arn:aws:sns:us-east-1:263340255179:NoStock:cd966da2-ba0e-4648-b5a2-7d63c31473bb</code>	Status Pending confirmation
Endpoint <code>binhngw@1067@fpt.edu.vn</code>	Protocol EMAIL
Topic NoStock	
Subscription Principal <code>arn:aws:iam::263340255179:role/vodlabs</code>	

Subscription filter policy | Redrive policy (dead-letter queue)

Subscription filter policy [info](#)
This policy filters the messages that a subscriber receives.

No filter policy configured for this subscription.
To apply a filter policy, edit this subscription.

Edit

Gmail

Compose

Inbox 14

Starred
Snoozed
Sent
Drafts
More

Labels

Search in emails

Active

1 of 252

AWS Notification - Subscription Confirmation

AWS Notifications <no-reply@sns.amazonaws.com> to me

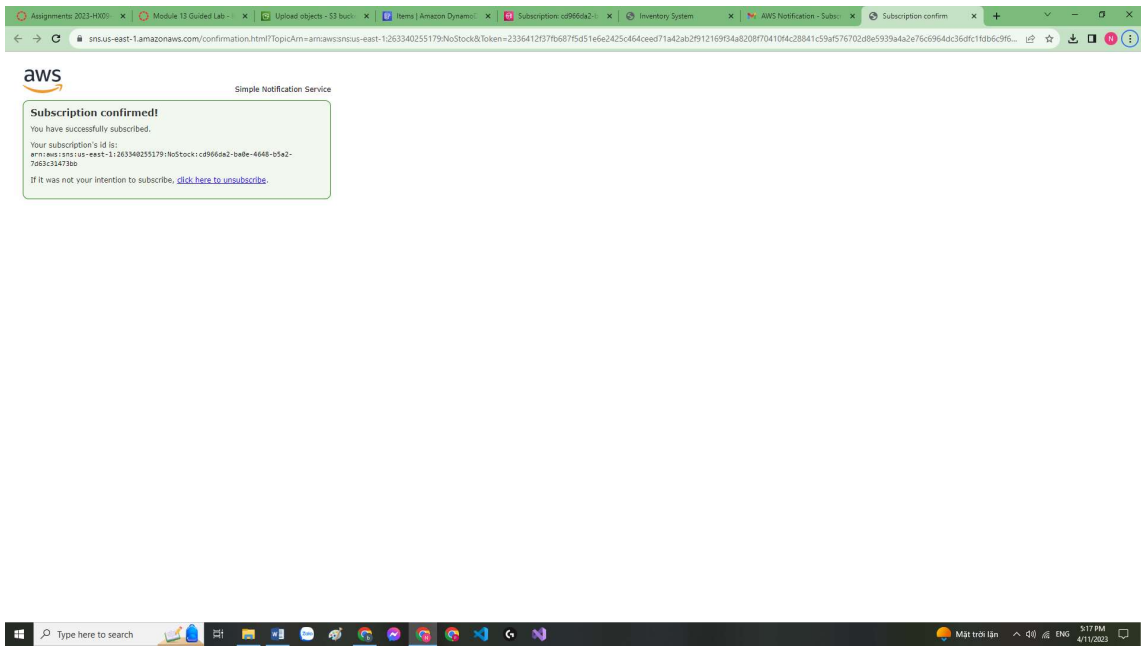
17:16 (3 minutes ago)

You have chosen to subscribe to the topic:
`arn:aws:sns:us-east-1:263340255179:NoStock`

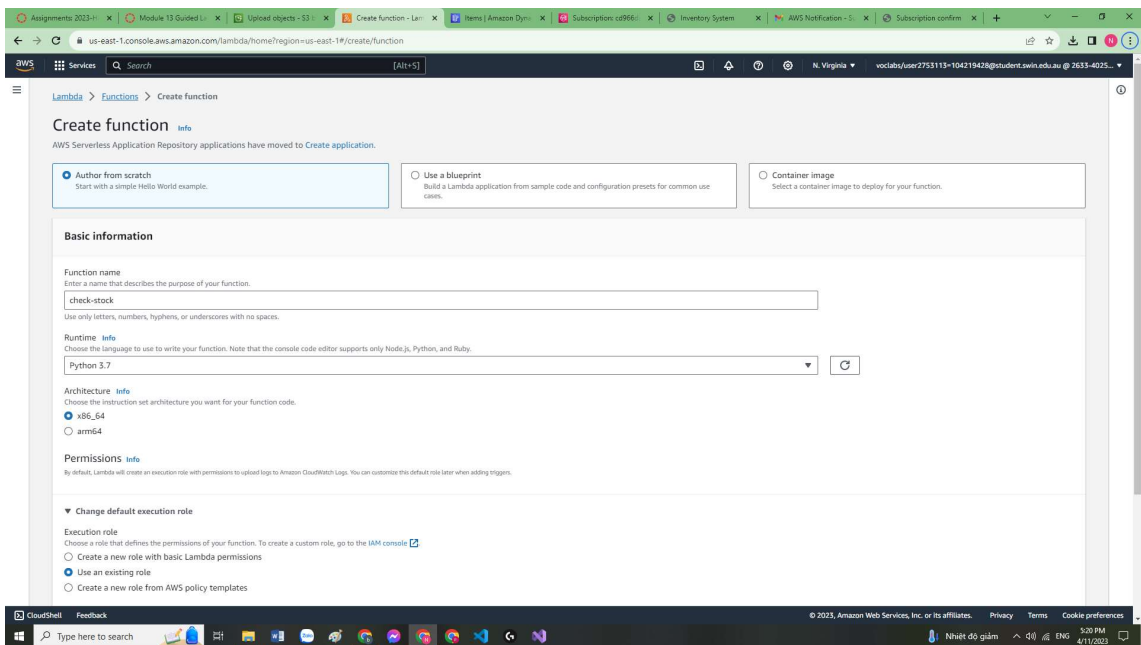
To confirm this subscription, click or visit the link below (if this was in error no action is necessary):
[Confirm subscription](#)

Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to [sns-not-out](#)

Reply Forward



Step 4: Create check-stock lambda function



Assignments 2023-1-1 x Module 13 Guided L... x Upload objects - S3 x Create function - Lan... x Items | Amazon Dyn... x Subscription: cd996... x Inventory System x AWS Notification - S... x Subscription confirm x

us-east-1 console.aws.amazon.com/lambda/home?region=us-east-1#/create/function

Services Search [Alt+S]

basic information

Function name
Enter a name that describes the purpose of your function.
check-stock
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.7

Architecture Info
Choose the instruction set architecture you want for your function code.
☒ x86_64
☐ arm64

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

☐ Create a new role with basic Lambda permissions
☒ Use an existing role
☐ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
Lambda-Check-Stock-Role
View the Lambda-Check-Stock-Role role on the IAM console.

► Advanced settings

Cancel Create function

CloudShell Feedback

Type here to search

20°C Nắng rải rác 5:20 PM 4/11/2023

us-east-1 console.aws.amazon.com/lambda/home?region=us-east-1#/functions/check-stock?function=true&tab=code

Services Search [Alt+S]

Successfully created the function check-stock. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > check-stock

check-stock Throttle Copy ARN Actions

▼ Function overview Info

check-stock Layers (0)

+ Add trigger + Add destination

Description
-
Last modified
4 seconds ago
Function ARN
arn:aws:lambda:us-east-1:263340255179:function:check-stock
Function URL Info

Code Test Monitor Configuration Aliases Versions

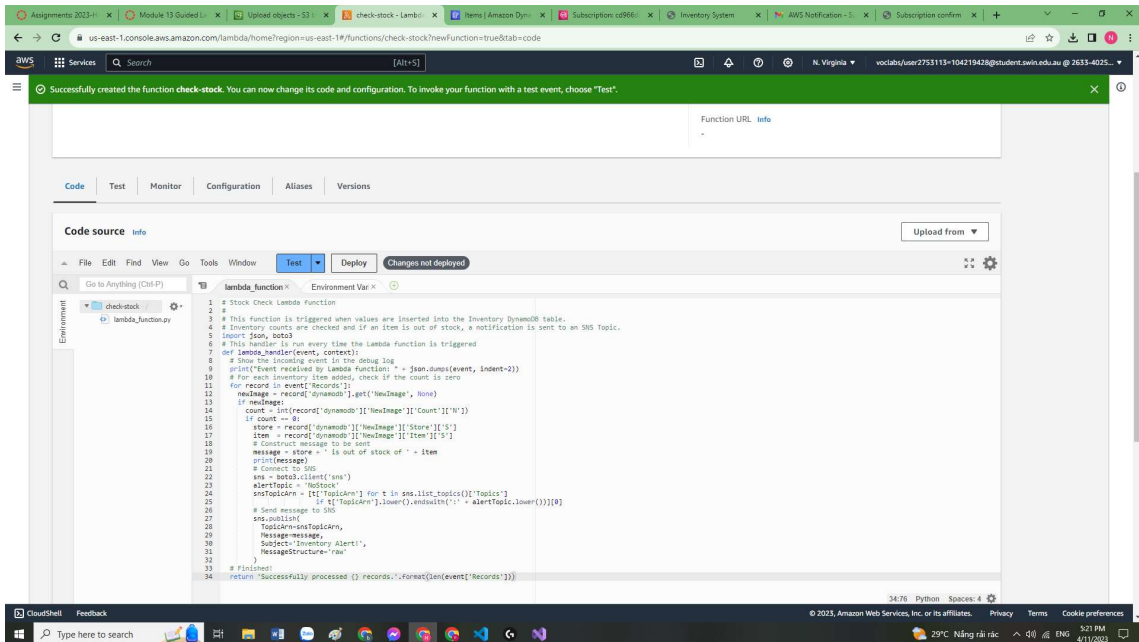
Code source Info Upload from

CloudShell Feedback

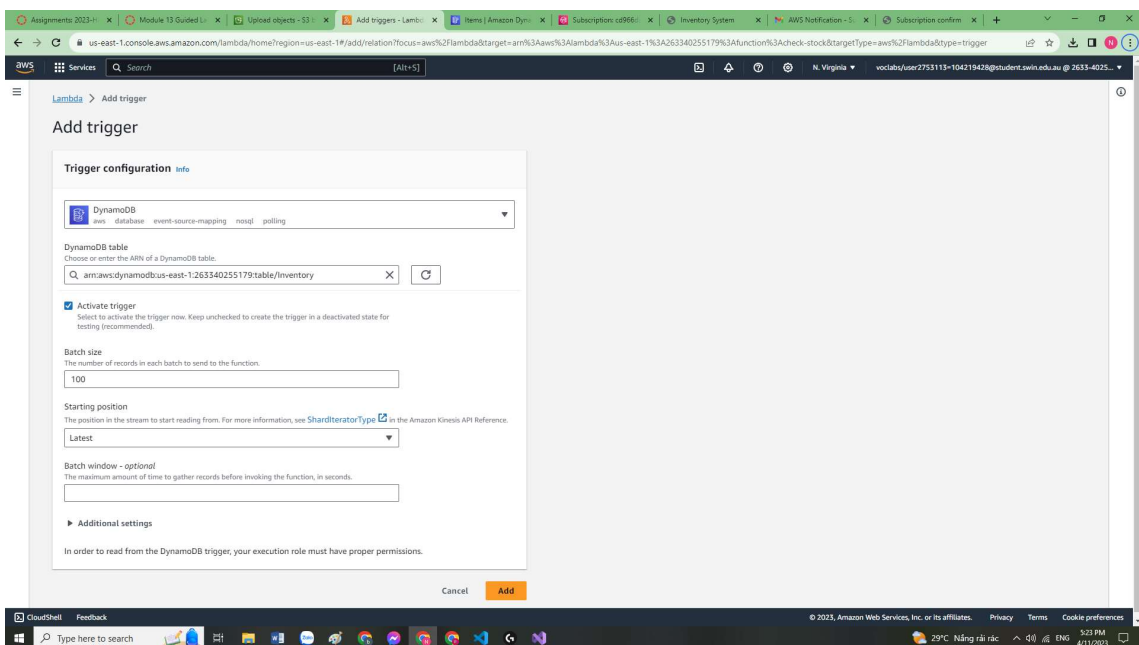
Type here to search

20°C Nắng rải rác 5:20 PM 4/11/2023

4.2: Use the check-stock code provided



4.3: add trigger



check-stock

Throttle Copy ARN Actions

The trigger inventory was successfully added to function check-stock. The trigger is in a disabled state.

Function overview

check-stock

Layers [0]

DynamoDB

+ Add destination

+ Add trigger

Description

Last modified 3 minutes ago

Function ARN amaws:lambda:us-east-1:263340255179:function:check-stock

Function URL

Code Test Monitor Configuration Aliases Versions

General configuration

Triggers

Permissions

Destinations

Function URL

Triggers (1)

Find triggers

Trigger

DynamoDB: inventory

amaws:dynamodb:us-east-1:263340255179:table/inventory/stream/2023-11-04T09:52:13.431

state: Enabled

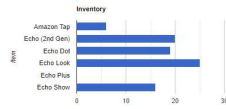
Fix errors Edit Delete Add trigger

aws-tc-largeobjects3-us-west-2.amazonaws.com/ILT-TF-200-ACACAD-20-EN/mod13-guided/web/inventory.htm?region=us-east-1&poolId=us-east-1-bba6b5f1-90df-4172-b82c-21607ce15283

Inventory Dashboard

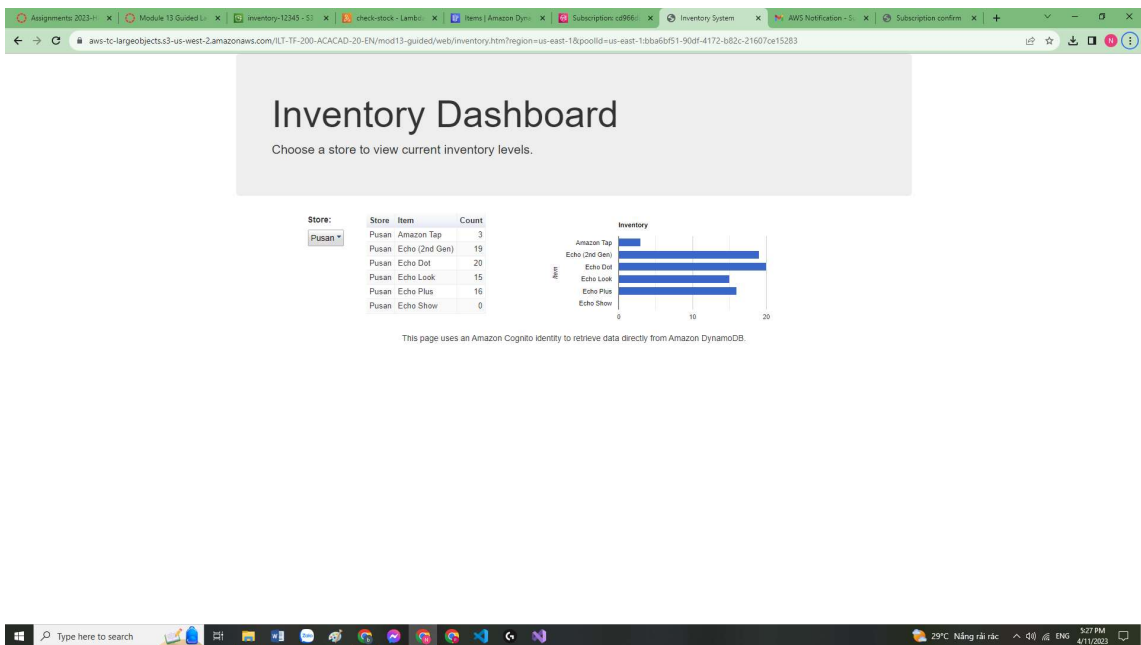
Choose a store to view current inventory levels.

Store:	Store	Item	Count
Karachi	Karachi	Amazon Tap	6
	Karachi	Echo (2nd Gen)	20
	Karachi	Echo Dot	19
	Karachi	Echo Look	25
	Karachi	Echo Plus	9
	Karachi	Echo Show	16



This page uses an Amazon Cognito Identity to retrieve data directly from Amazon DynamoDB.

29°C Nắng rải rác 5:28 PM 4/11/2023



And so on

After 10 minutes there were no notification so I'll end it here