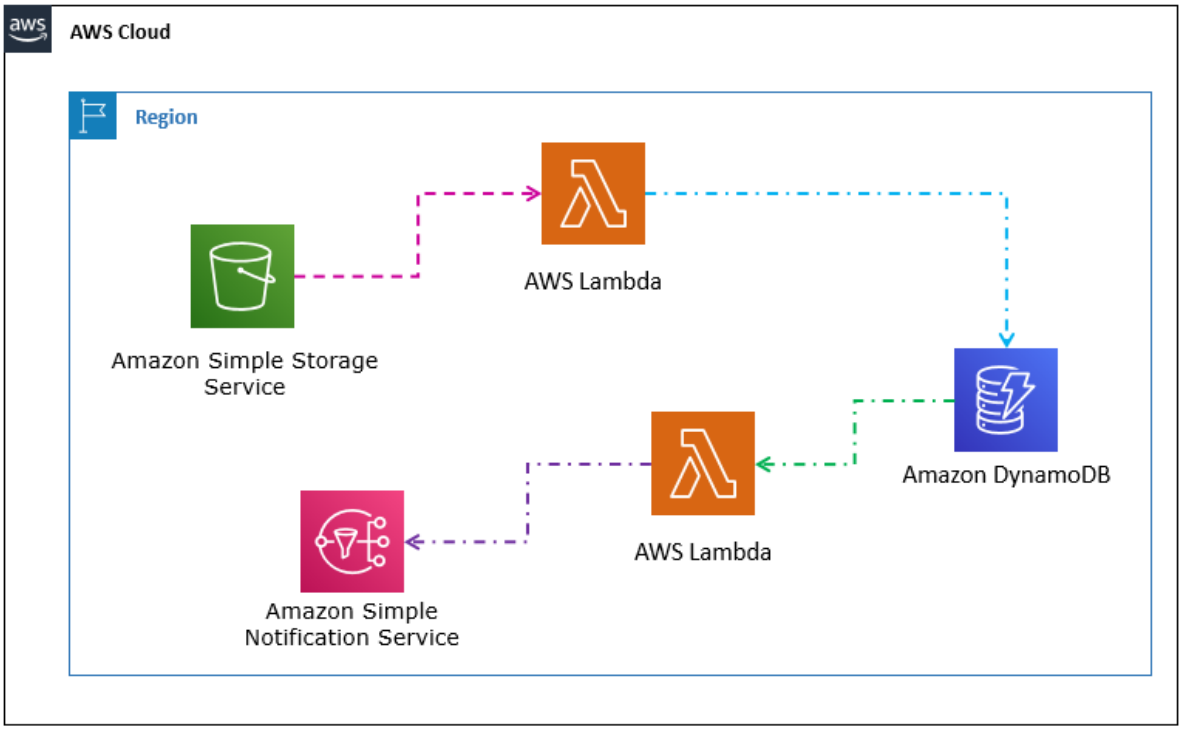
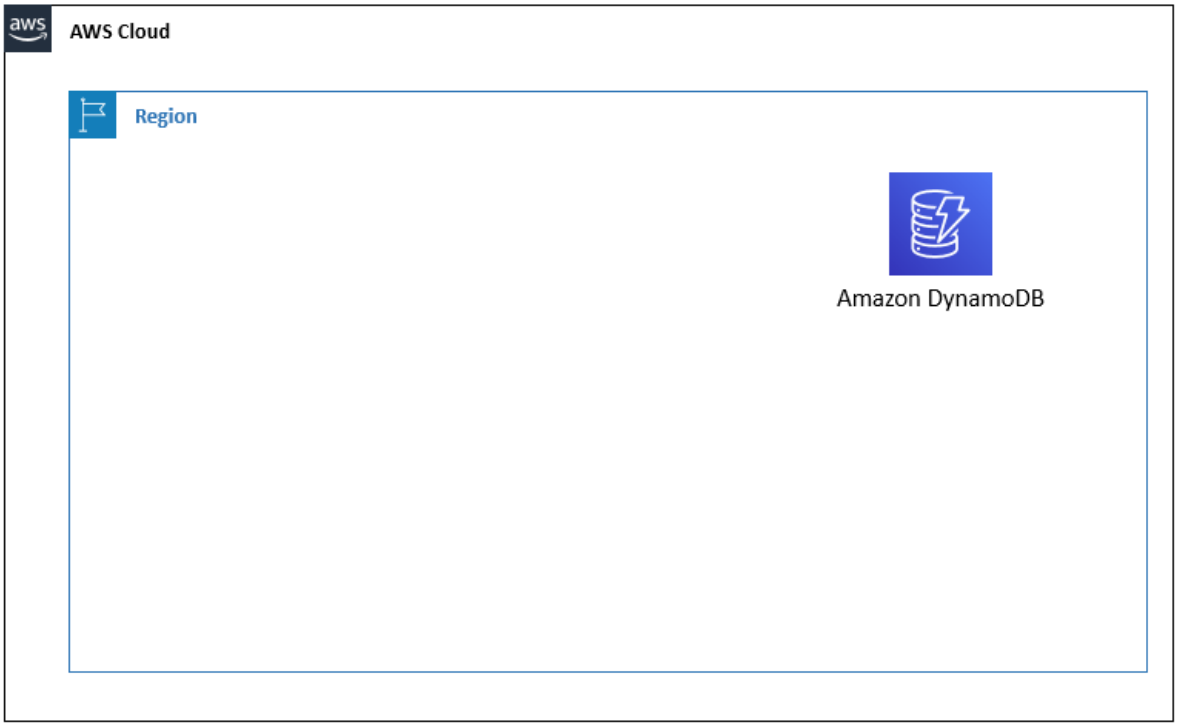
Implementing a Serverless Architecture

(LAB-11)



Task 1: Create DynamoDB Table



Step 1: Choose a Region Using the Console

Choose the **US East (N. Virginia)** region list to the right of your account information on the navigation bar.

Step 2: Create a DynamoDB Table

1. In the **AWS Management Console**, on the **Services** menu, click **DynamoDB**

2. Choose **Create Table** and configure:

    a. **Table** name: Write **Inventory**

    b. **Primary key**:

    i. **Partition key**: Write **Store**

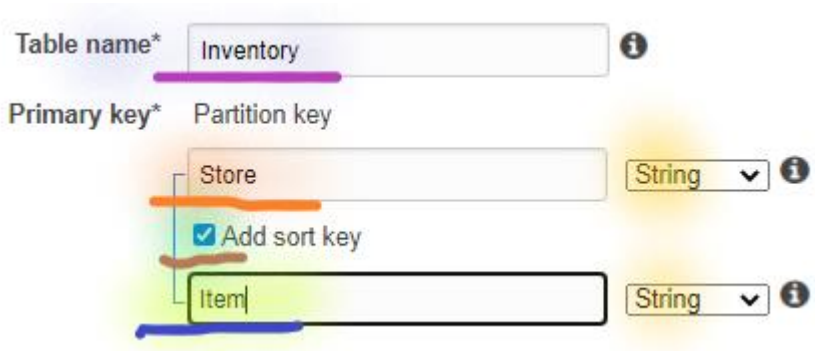
    - Set the data type to **String**

    ii. Select the **Add sort key**

    iii. In the box, write **Item**

    - Set the data type to **String**

Note: Leave other details as default.



iv. Select **Create**

Note: Deployment will take few mnts. No need to wait, **go to the next task**.

Task 2: Create IAM Role

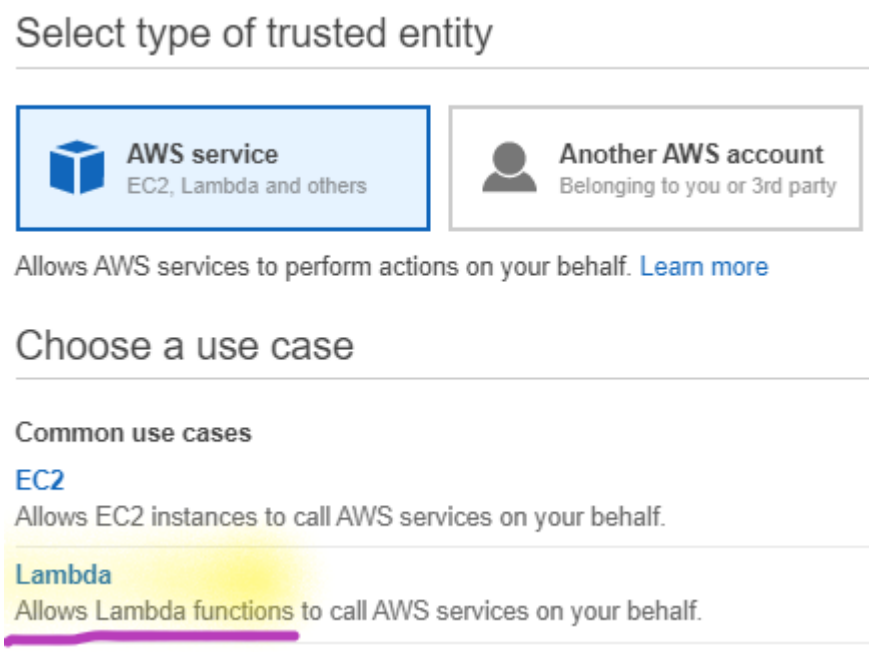
Step 1: Create Two IAM Roles for AWS Lambda

1. In the **AWS Management Console**, on the **Services** menu, click **IAM**

Create First Role

2. Select **Roles**, click on **Create role**

   a. Select **Lambda**



   b. Select **Next: Permissions**

   c. Search and Select **AmazonDynamoDBFullAccess**

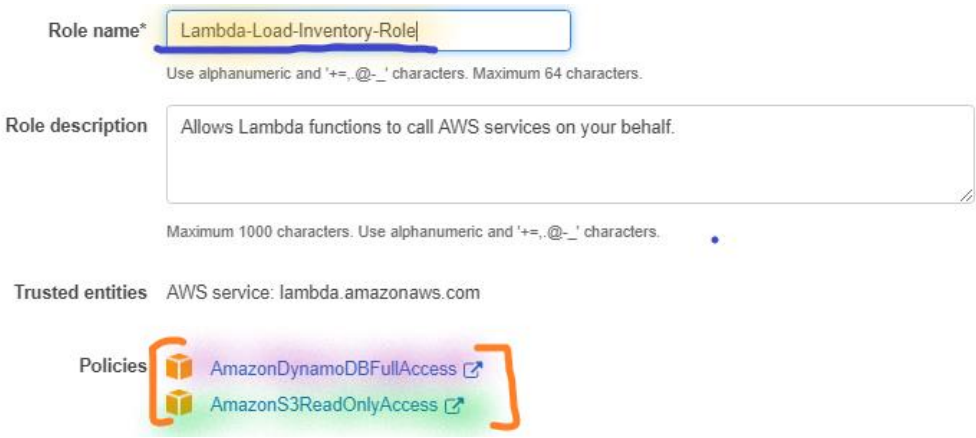
   d. Again, Search and Select **AmazonS3ReadOnlyAccess**

e. Select **Next: Tags**

f. Select **Next: Review**

Note: You will see **DynamoDBFullAccess** and **S3ReadOnlyAccess** under policies.

g. **Role Name**: Write **Lambda-Load-Inventory-Role**



h. Click **Create role**

Note: Wait for role creation. Once created you get the message The role **lambda-load-inventory-role** has been created.

Create Second Role

3. Select **Roles**, click on **Create role**

   a. Select **Lambda**, select **Next: Permissions**

   b. Search and Select **AmazonSNSFullAccess**

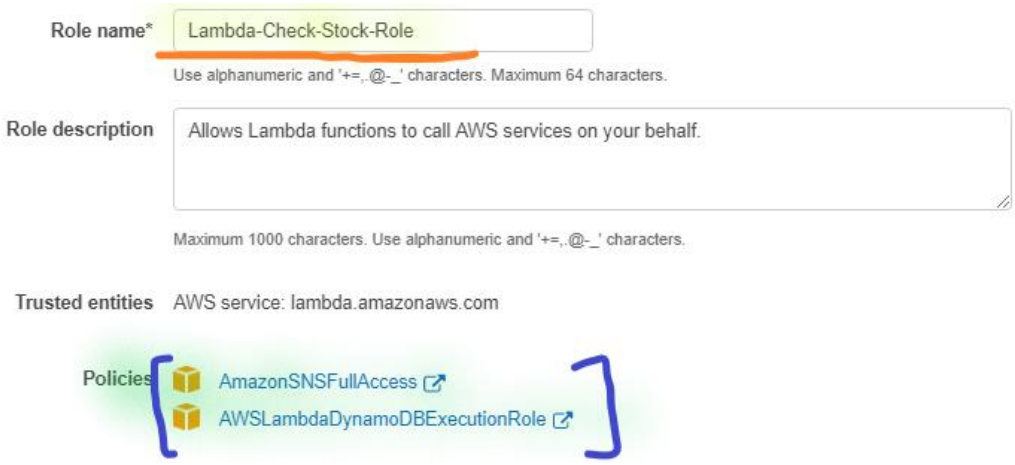
   c. Again, Search and Select **AWSLambdaDynamoDBExecutionRole**

   d. Select **Next: Tags**

e. Select **Next: Review**

Note: You will see **AmazonSNSFullAccess** and **AWSLambdaDynamoDBExecutionRole** under policies.

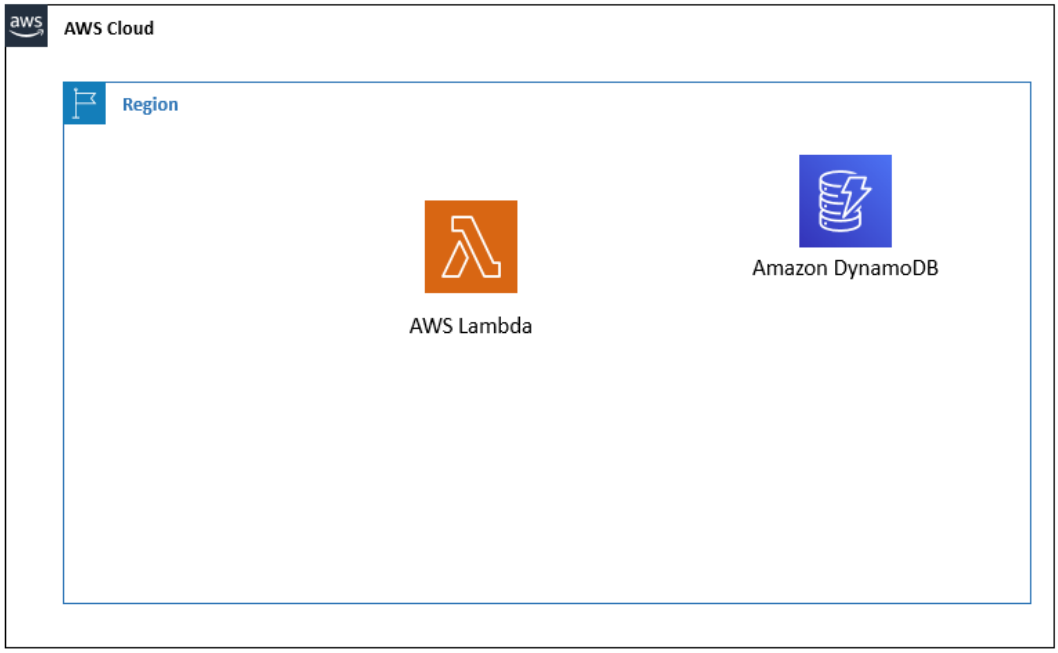
f. **Role name**: Write **Lambda-Check-Stock-Role**



g. Click **Create role**

Note: Wait for role creation. Once created you get the message The role **lambda-check-stock-role** has been created.

Task 3: Create Lambda Function to Store Data in DynamoDB



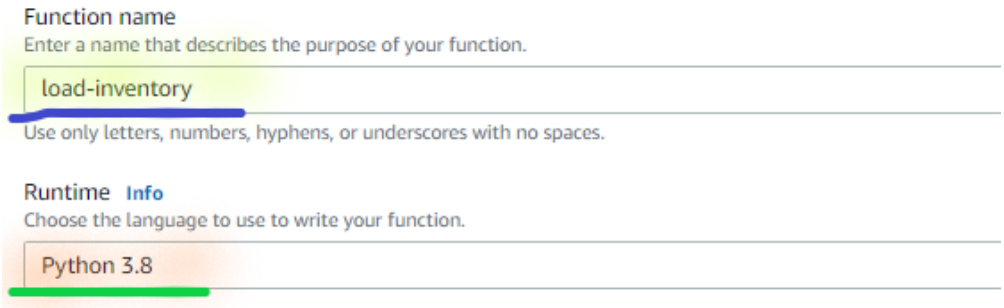
Step 1: Create a Lambda Function to Load Data

In this task, you will create **an AWS Lambda function** that will process an inventory file. The Lambda function will read the file and insert information into an Amazon DynamoDB table.

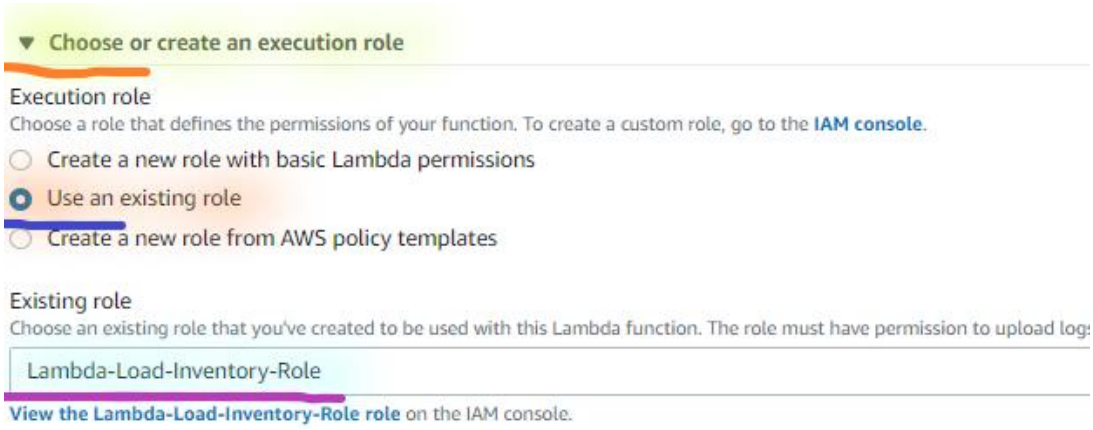
1. In the **AWS Management Console**, on the **Services** menu, click **Lambda**.
2. Click **Create a function**

Blueprints are code templates for writing Lambda functions. Blueprints are provided for standard Lambda triggers. This lab provides you with a pre-written Lambda function, so you will Author from scratch.

3. Select **Author from scratch** and configure:
   i. **Name**: Write **load-inventory**
   ii. **Runtime**: Dropdown and Select **Python 3.8**



   iii. **Expand** Choose or create an execution role
   iv. **Role**: Select **Choose an existing role**
      - **Existing role**: Dropdown and Select **Lambda-Load-Inventory-Role**



This role gives execution permissions to the Lambda function so it can access Amazon S3 and Amazon DynamoDB.

4. Click **Create function**

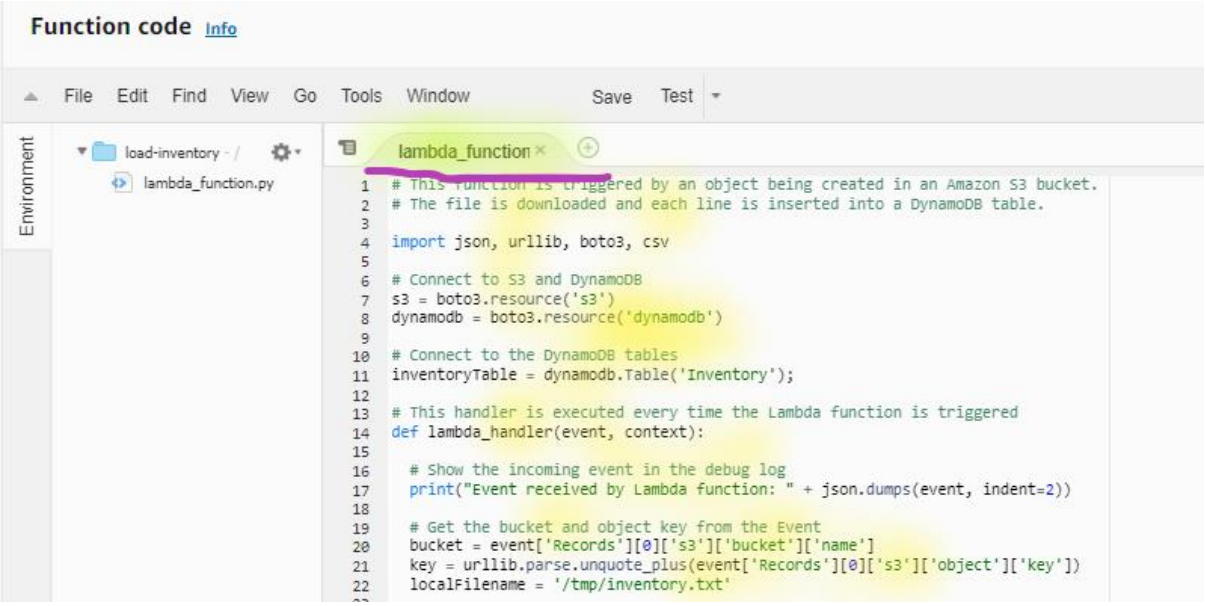
5. Once function gets successfully created, scroll down to the **Function code** section, then **delete** all the that appears in the code editor.

6. Copy and paste the code into the **Function code** editor from **Lambdafunction-1.txt** file.

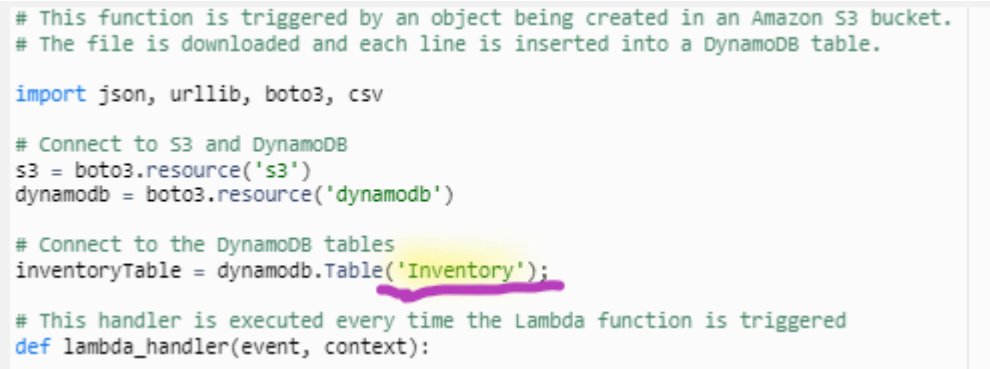
Note: Code for **Lambdafunction-1** is available with the lab manual.

#**Examine the code**. It is performing the following steps:

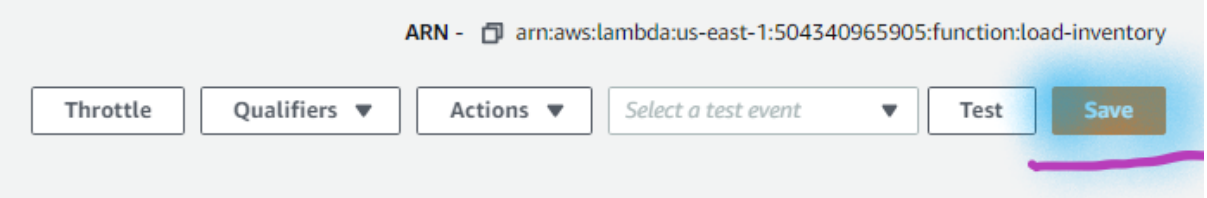
- Download the file from Amazon S3 that triggered the event
- Loop through each line in the file
- Insert the data into the DynamoDB *Inventory* table



7. Ensure that the **DynamoDB table** name must be same as you have created in the previous step.



8. Click **Save** at the top left side of the page.

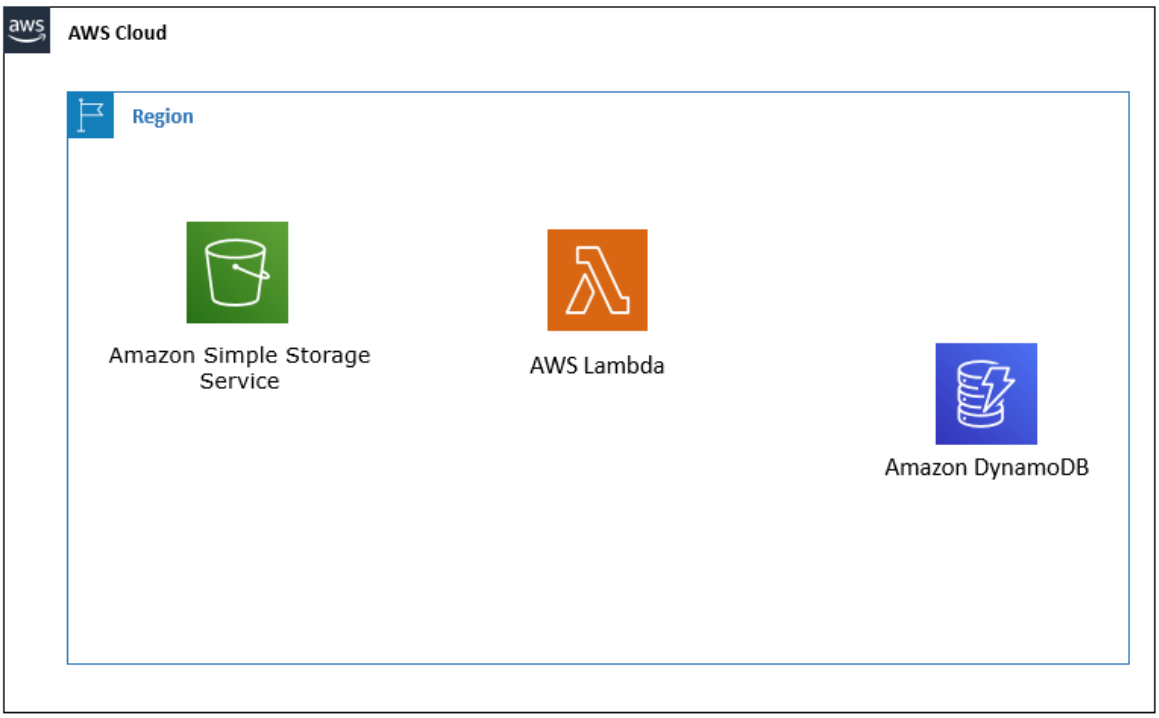


Next, you will configure Amazon S3 to trigger the Lambda function when a file is uploaded.

Step 2: Configure an Amazon S3 Event

Stores from around the world will provide inventory files to load into the inventory tracking system. Rather than uploading files via FTP, the stores can upload directly to Amazon S3. This can be done via a web page, a script or as part of a program. Once a file is received, the AWS Lambda function will be triggered, and it will load the inventory into a DynamoDB table.

In this task you will create an Amazon S3 bucket and configure it to trigger the Lambda function.



1. In the **AWS Management Console**, on the **Services** menu, click **S3**.
2. Click **Create bucket**
    a. **Bucket name**: Write **inventory-123**
    b. **Region**: Dropdown and Select **US East (N. Virginia)**

Note: Replace 123 to make the bucket name unique.

Note: Leave other details as default.

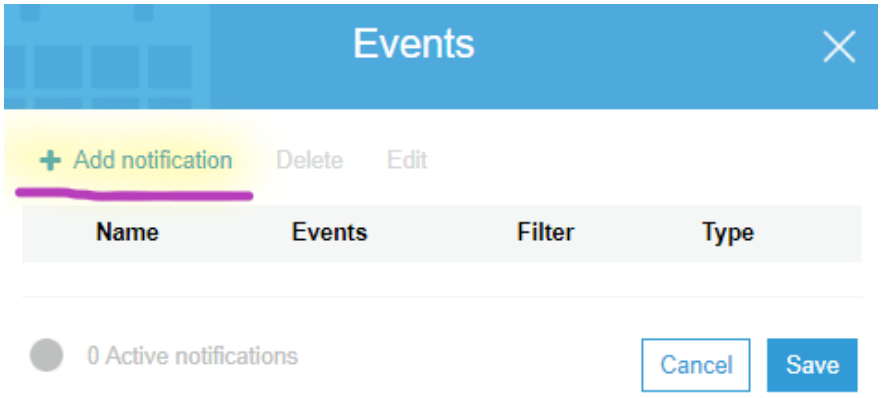
c. Click **Create**

You will now configure the bucket to automatically trigger the Lambda function whenever a file is uploaded.

3. Open the **inventory-123** bucket.
4. Click the **Properties** tab.
5. Scroll down to **Advanced settings**, then click **Events**.

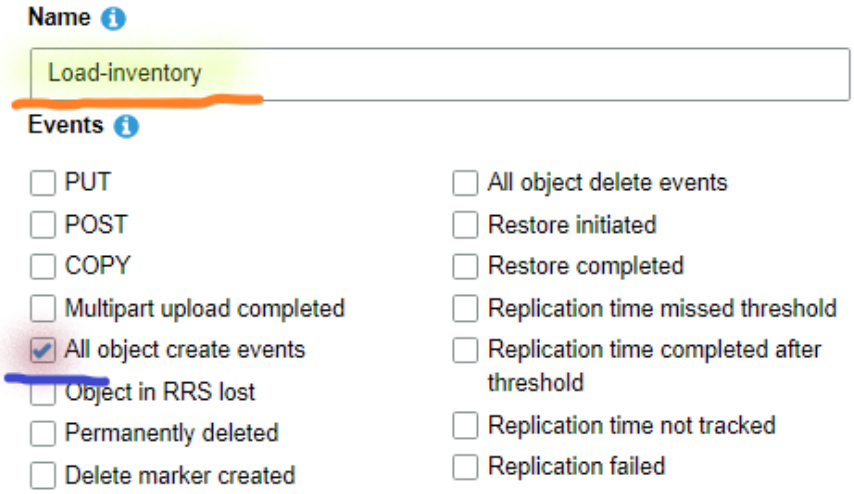
You will configure an event to trigger when an object is created in the S3 bucket.

6. Click **Add notification** then **configure**:



a. **Name**: Write **Load-inventory**

b. **Events**: Select **All object create events**



Note: Ensure Only All object create events should be selected.

c. **Send to**: Dropdown & select **Lambda Function**

d. **Lambda**: Dropdown & select **Load-Inventory**

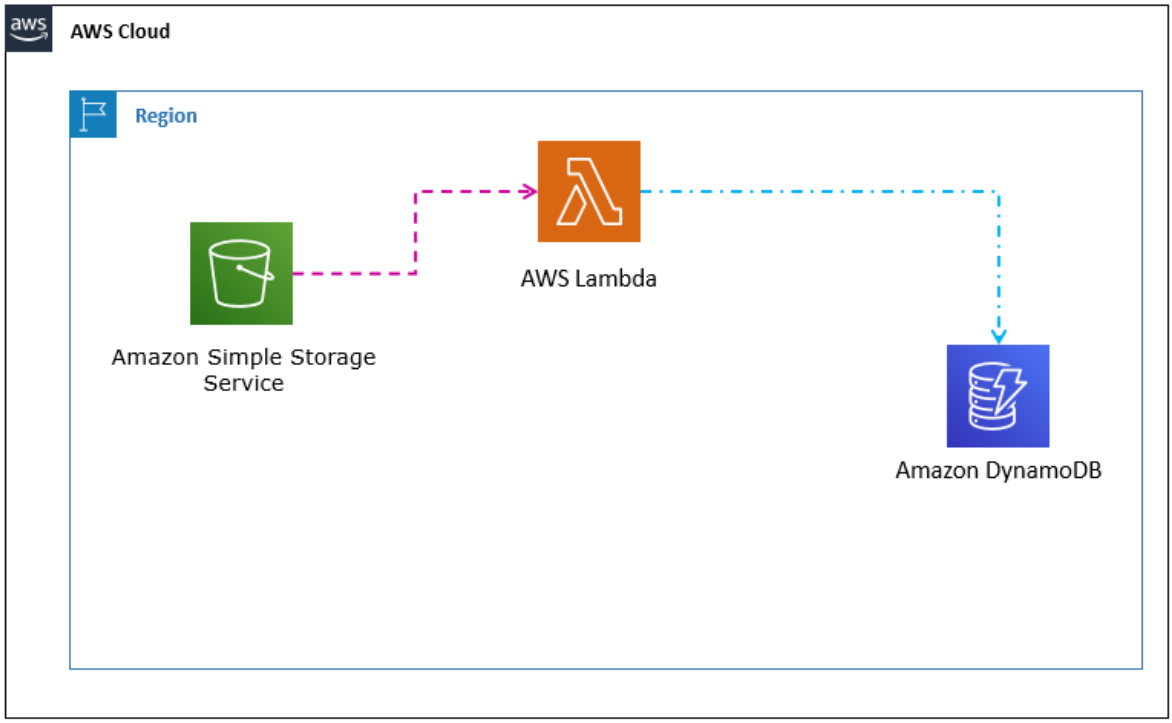
e. Click **Save**

This will tell Amazon S3 to trigger the *Load-Inventory* Lambda function you created earlier whenever an object is created in the bucket.

Note: Your bucket is now ready to receive inventory files!

Step 3: Test the Loading Process

You are now ready to test the loading process. You will upload an inventory file, then check that it loaded successfully.



1. **Unzip** the **inventory-files.zip** file

The zip file contains multiple inventory CSV files that you can use to test the system. Here are the contents of the **Berlin file**:

Store	Item	Count
Berlin	Echo Dot	12
Berlin	Echo (2nd Gen)	19
Berlin	Echo Show	18
Berlin	Echo Plus	0
Berlin	Echo Look	10
Berlin	Amazon Tap	15

2. In the **AWS Management Console**, on the **Services** menu, click **S3**.
3. Select **inventory-123** bucket.
4. Click **Upload** and upload one of the CSV files to the bucket. (You can choose any of the inventory files.)

Note: Amazon S3 will automatically trigger the Lambda function, which will load the data into a DynamoDB table.

You can also view the data within the DynamoDB table.

5. In the **AWS Management Console**, on the **Services** menu, click **DynamoDB**.
6. Click **Tables**.
7. Open the **Inventory** table.
8. Click the **Items** tab.

Note: You can view the data from the inventory file will be displayed, showing the Store, Item and inventory Count.

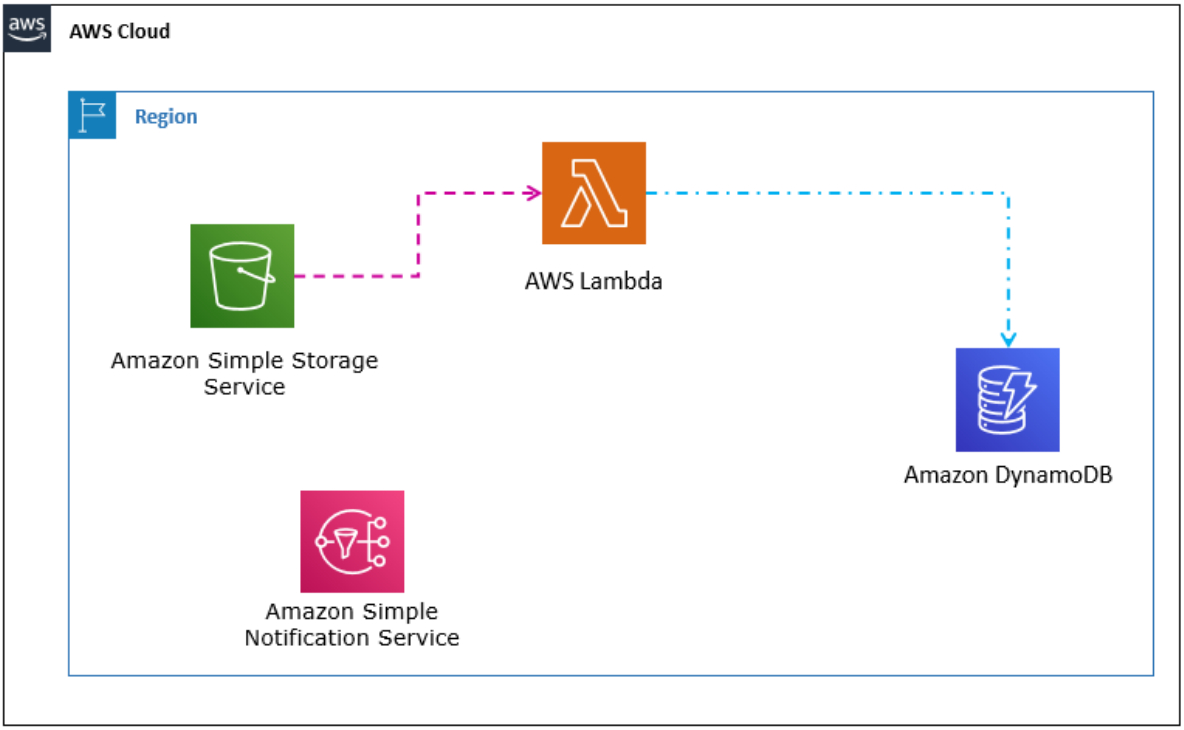
Task 4: Create Lambda Function to Send Email Notification

Step 1: Configure Notifications

You wish to notify inventory management staff when a store runs out of stock of an item. For this serverless notification functionality, you will use **Amazon Simple Notification Service (SNS)**.

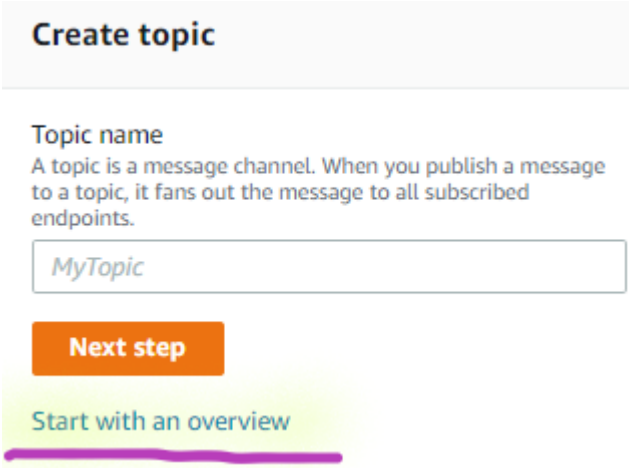
Amazon SNS is a flexible, fully managed publish/subscribe messaging and mobile notifications service for the delivery of messages to subscribing endpoints and

clients. With SNS you can fan-out messages to a large number of subscribers, including distributed systems and services, and mobile devices.



1. In the **AWS Management Console**, on the **Services** menu, click **Simple Notification Service**

2. Click **Start with overview**

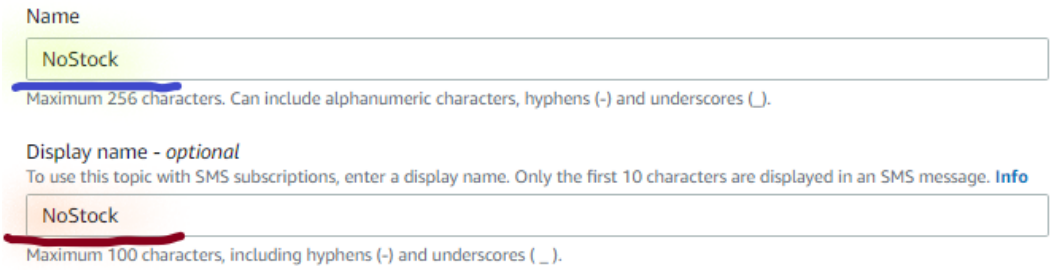


3. Click **Topics**

4. Click **Create topic** and configure:

   i. **Topic name**: Write **NoStock**

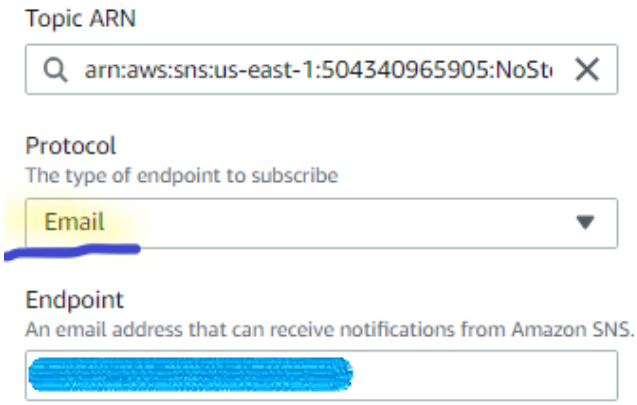
   ii. **Display name**: Write **NoStock**



iii. Click **Create topic**

To receive notifications, you must **subscribe** to the Topic. You can choose to receive notifications via several methods, such as SMS and email.

5. Select the **Topics**
6. Open the **NoStock** topics.
7. Click **Create subscription** and configure:
   i. **Protocol**: Dropdown and Select **Email**
   ii. **Endpoint**: Write **your email id**



iii. Click **Create subscription**

Note: After creating an Email subscription, a confirmation email will be sent to you. Open the email and click the *Confirm subscription* link.

8. You will also receive email for subscription confirmation. **Confirm the subscription**

You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:504340965905:NoStock

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

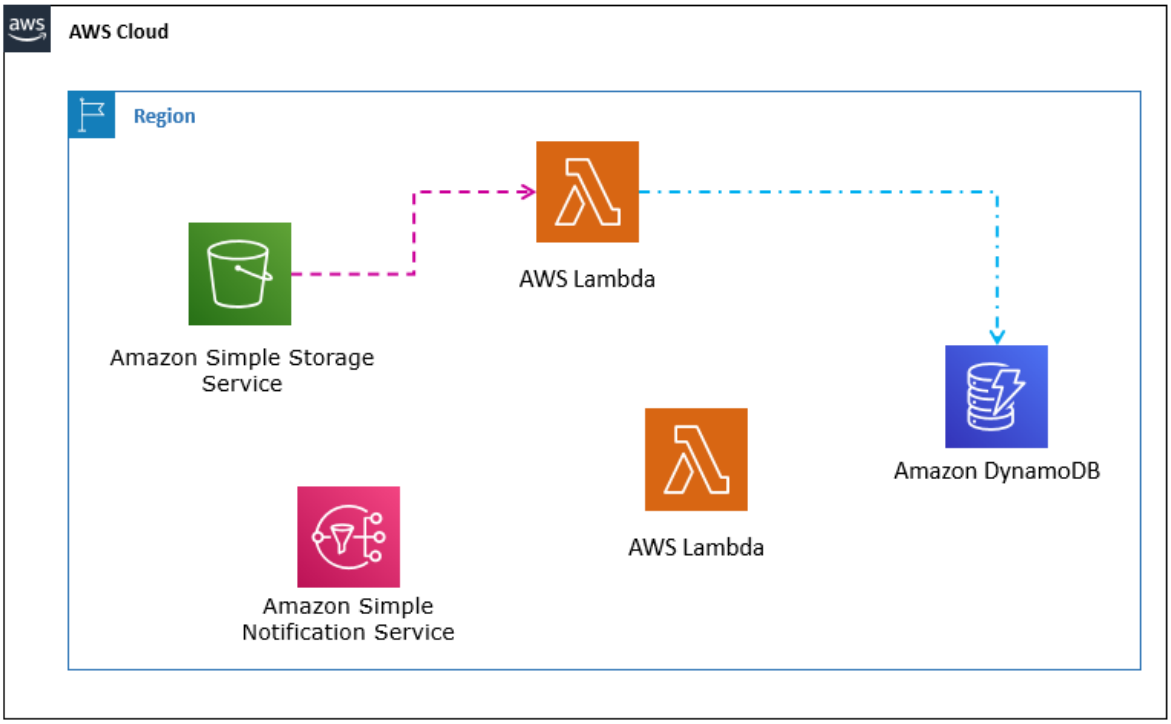
Step 2: Create a Lambda Function to Send Notifications

While you could modify the existing *Load-Inventory* Lambda function to check inventory levels while the file is being loaded, this is not a good architectural practice. Rather than overloading the *Load-Inventory* function with business logic, you will create another Lambda function that is triggered whenever data is loaded into the DynamoDB table. This will be triggered by a *DynamoDB Stream*.

There are several benefits to this architectural approach:

- Each Lambda function performs a single, specific function. This makes the code simpler and more maintainable.
- Additional business logic can be added by creating additional Lambda functions. Each function operates independently, so existing functionality is not impacted.

In this task, you will create another Lambda function that looks at inventory as it is loaded into the DynamoDB table. If it notices that an item is Out of Stock, it will send a notification via the Amazon SNS topic you created earlier.



1. In the **AWS Management Console**, on the **Services** menu, click **Lambda**.
2. Click **Create a function**
3. Select **Author from scratch** and configure:
    i. **Name**: Write **check-stock**
    ii. **Runtime**: Dropdown and Select **Python 3.8**
    iii. **Expand** Choose or create an execution role
    iv. **Role**: Select **Choose an existing role**
        - **Existing role**: Dropdown and Select **Lambda-Check-Stock-Role**

This role has been configured with permissions to send a notification to Amazon SNS.

4. Click **Create function**
5. Once function gets successfully created, scroll down to the **Function code** section, then **delete** all the that appears in the code editor.
6. Copy and paste the code into the **Function code** editor from **Lambdafunction-2.txt** file.

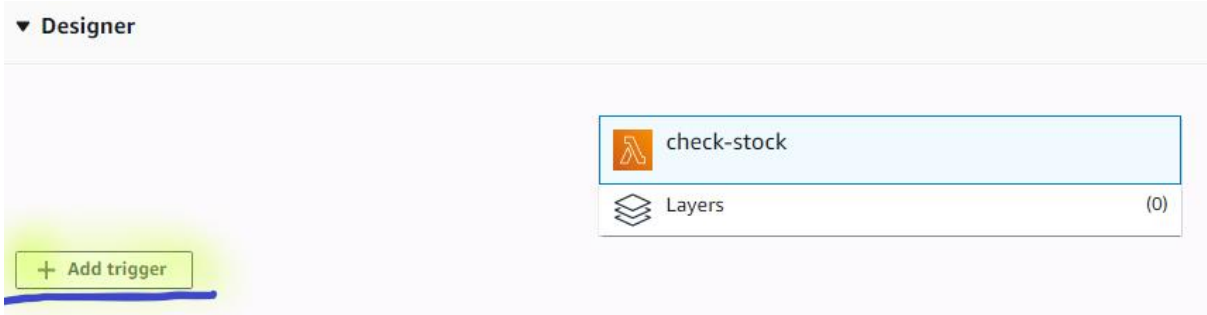
Note: Code for **Lambdafunction-2** is available with the lab manual.

#**Examine the code**. It is performing the following steps:

- Loop through the incoming records
- If the inventory count is zero, send a message to the ***NoStock*** SNS topic

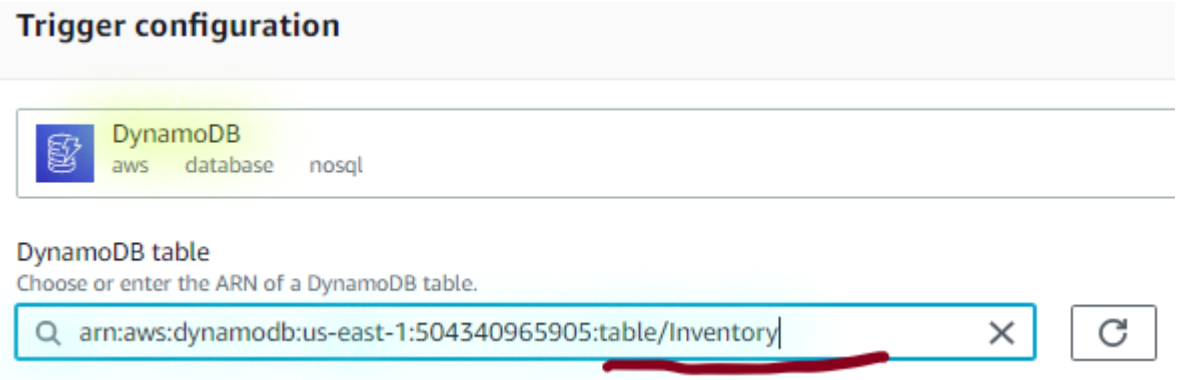
You will now configure the function to be triggered whenever data is added to the *Inventory* table in DynamoDB.

7. Click **Save** at the top left side of the page.
8. Scroll up and Select **Add triggers**.



9. Scroll down to the **Configure triggers** section and configure:

    i. **DynamoDB Table**: Dropdown & select **Inventory**



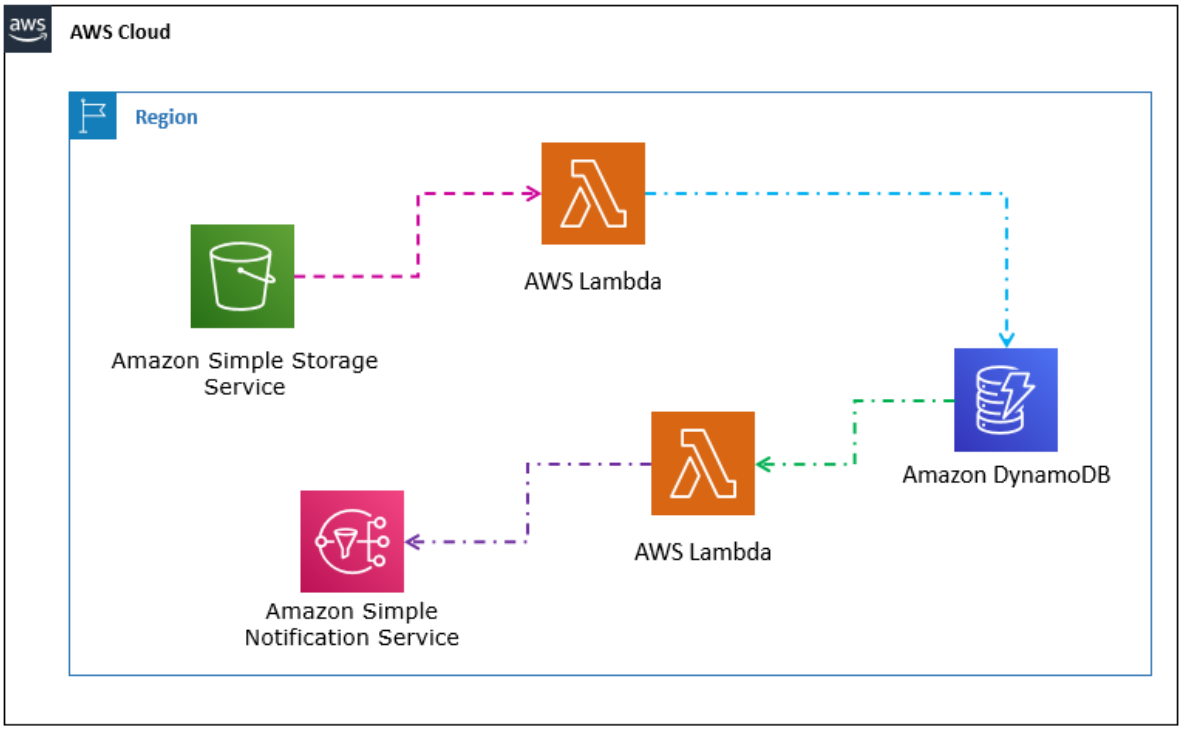
    ii. Click **Add**

10. Click **Save** at the top of the page.

Note: You are now ready to test the system!

Step 3: Test the System

You will now upload an inventory to Amazon S3, which will trigger the original *Load-Inventory* function. This function will load data into DynamoDB, which will then trigger the new *Check-Stock* Lambda function. If the Lambda function detects an item with zero inventory, it will send a message to Amazon SNS, which will notify you via Email.



1. In the **AWS Management Console**, on the **Services** menu, click **S3**.
2. Select **inventory-123** bucket.
3. Click **Upload** and upload one of the CSV files to the bucket. (You can choose any of the inventory files, different then what uploaded earlier.)

You should receive a **notification via Email** telling you that the store is out of stock of an item (every inventory file has one item out-of-stock).

Note: If you did not receive a notification, please wait a few minutes and try uploading a different inventory file. The DynamoDB trigger may sometimes take a few minutes to enable.

Task 5: Delete Environment

Step 1: Delete Environment

1. Delete **S3** bucket
2. Delete **DynamoDB** Table
3. Delete **Lambda** Functions