Marwadi University Marwadi Chandarana Group	**Marwadi University** **Faculty of Engineering and Technology** **Department of Information and Communication Technology**	
Subject: Cloud Developing (01CT0720)	**Aim: Create Lambda functions using the AWS SDK for Python.**	
Experiment No: 06	**Date:**	**Enrolment No: 92200133030**

Aim :- Create Lambda functions using the AWS SDK for Python.

Lab overview and objectives

In this lab, you will use the AWS SDK for Python (boto3) to create AWS Lambda functions. Calls to the REST API that you created in the earlier Amazon API Gateway lab will initiate the functions. One of the Lambda functions will perform either an Amazon DynamoDB database table scan or an index scan. Another Lambda function will return a standard acknowledgment message that you will enhance later in a lab where implement Amazon Cognito.

After completing this lab, you should be able to:

- Create a Lambda function that queries a DynamoDB database table.
- Grant sufficient permissions to a Lambda function so that it can read data from DynamoDB.
- Configure REST API methods to invoke Lambda functions using Amazon API Gateway.

AWS service restrictions

In this lab environment, access to AWS services and service actions might be restricted to the ones that are needed to complete the lab instructions. You might encounter errors if you attempt to access other services or perform actions beyond the ones that are described in this lab.

Scenario

The café is eager to launch a dynamic version of their website so that the website can access data stored in a database. Sofía has been making steady progress toward this goal.

In a previous lab, you played the role of Sofía and created a DynamoDB database. The database table contains café menu details, and an index holds menu items that are flagged as specials. Then, in another lab, you created an API to add the ability for the website to receive mock data through REST API calls.

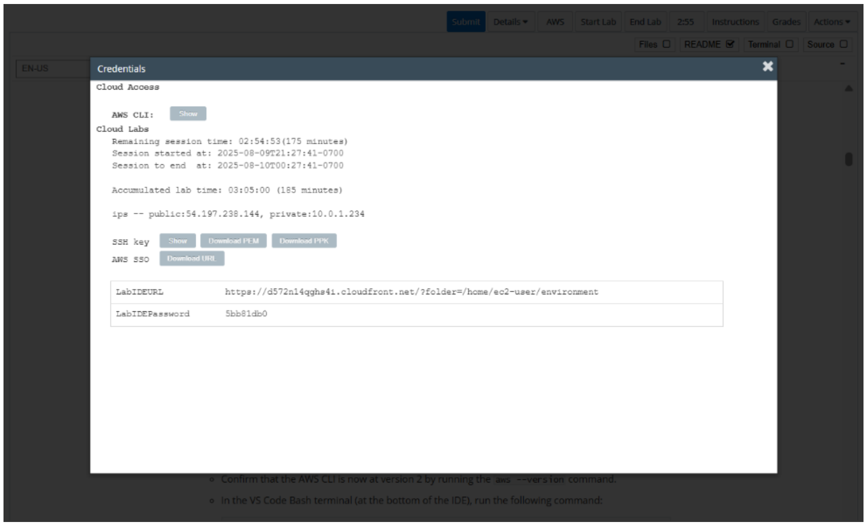
In this lab, you will again play the role of Sofía. You will replace the mock endpoints with functional endpoints so that the web application can connect to the database. You will use Lambda to bridge the connection between the GET APIs and the data stored in DynamoDB. Finally, for the POST API call, Lambda will return an updated acknowledgment message.

Task 1: Preparing the lab

Connect to the VS Code IDE.

1. At the top of these instructions, choose Details followed by **AWS: Show**
2. Copy values from the table **similar** to the following and paste it into an editor of your choice for use later.
   a. **LabIDEURL**

b. **LabIDEPassword**

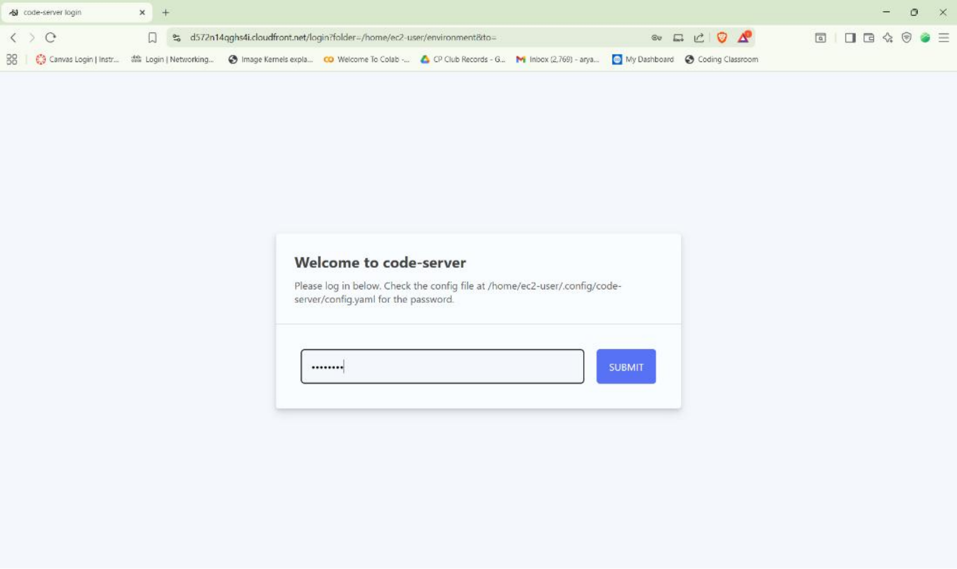


```
Cloud Access

   AWS CLI:
Cloud Labs
   Remaining session time: 02:54:53(175 minutes)
   Session started at: 2025-08-09T21:27:41-0700
   Session to end  at: 2025-08-10T00:27:41-0700

   Accumulated lab time: 03:05:00 (185 minutes)

   ips -- public:54.197.238.144, private:10.0.1.234

   SSH key
   AWS SSO
```

LabIDEURL	https://d572n14qghs4i.cloudfront.net/?folder=/home/ec2-user/environment
LabIDEPassword	5bb81db0

- Confirm that the AWS CLI is now at version 2 by running the `aws --version` command.
- In the VS Code Bash terminal (at the bottom of the IDE), run the following command:

3. In a new browser tab, paste the value for **LabIDEURL** to open the VS Code IDE.
4. On the prompt window **Welcome to code-server**, enter the value for **LabIDEPassword** you copied to the editor earlier, choose **Submit** to open the VS Code IDE.

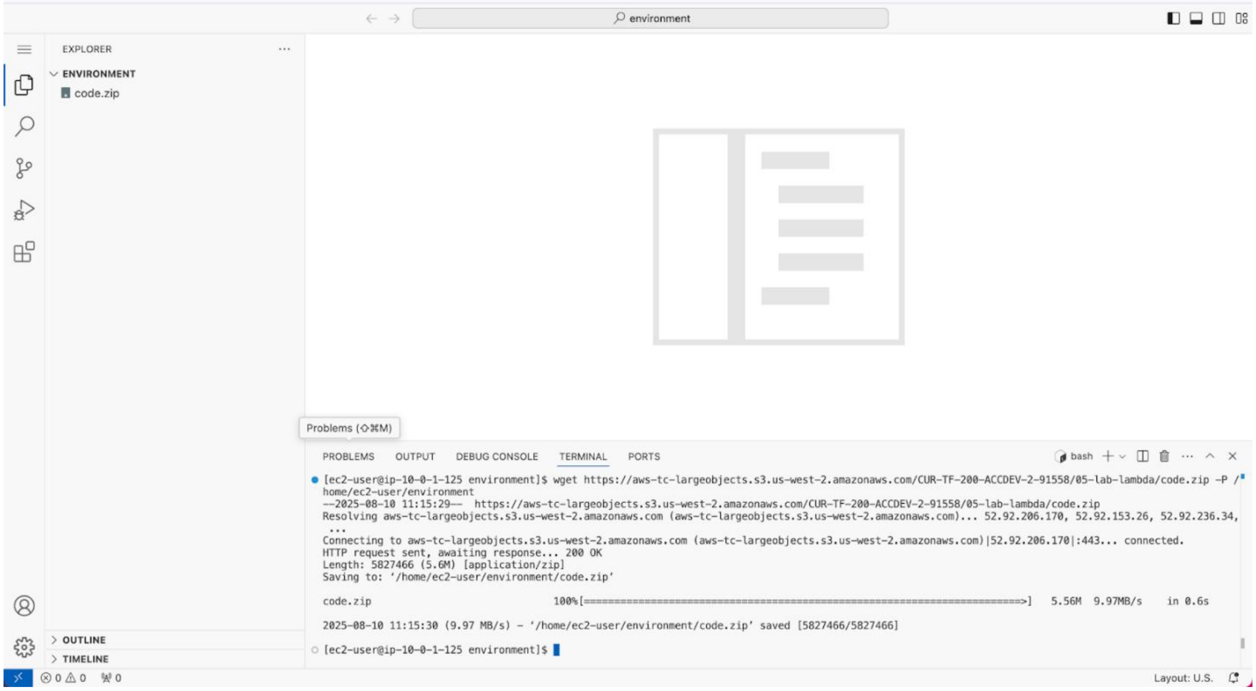


Welcome to code-server

Please log in below. Check the config file at /home/ec2-user/.config/code-server/config.yaml for the password.

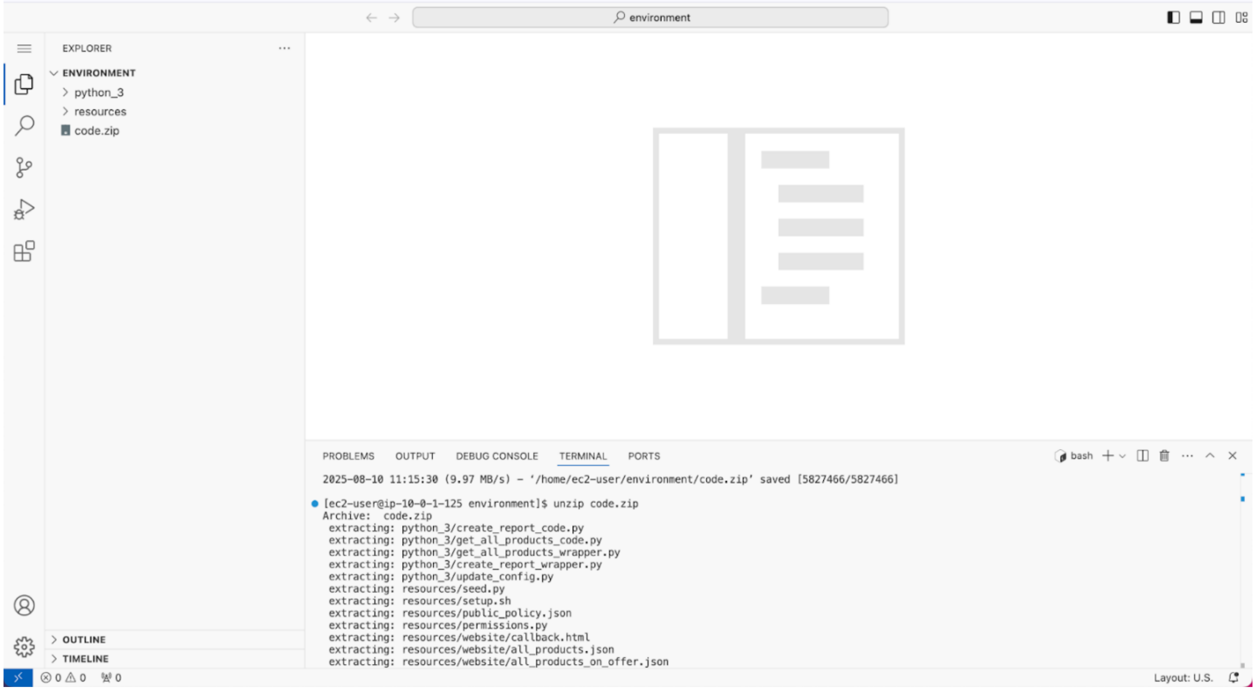
5. Download and extract the files that you need for this lab.
   - In the VS Code bash terminal (located at the bottom of the IDE), run the following commands:

```
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCDEV-2-91558/05-lab-lambda/code.zip -P /home/ec2-user/environment
```



```
[ec2-user@ip-10-0-1-125 environment]$ wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCDEV-2-91558/05-lab-lambda/code.zip -P /
home/ec2-user/environment
--2025-08-10 11:15:29--  https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCDEV-2-91558/05-lab-lambda/code.zip
Resolving aws-tc-largeobjects.s3.us-west-2.amazonaws.com (aws-tc-largeobjects.s3.us-west-2.amazonaws.com)... 52.92.206.170, 52.92.153.26, 52.92.236.34,
 ...
Connecting to aws-tc-largeobjects.s3.us-west-2.amazonaws.com (aws-tc-largeobjects.s3.us-west-2.amazonaws.com)|52.92.206.170|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5827466 (5.6M) [application/zip]
Saving to: '/home/ec2-user/environment/code.zip'

code.zip                   100%[=================================================================================>]   5.56M  9.97MB/s    in 0.6s

2025-08-10 11:15:30 (9.97 MB/s) - '/home/ec2-user/environment/code.zip' saved [5827466/5827466]

[ec2-user@ip-10-0-1-125 environment]$ 
```

6. You should see that the **code.zip** file was downloaded to the VS Code IDE and is now in the left navigation pane.
   - Extract the file by running the following command:
     `unzip code.zip`

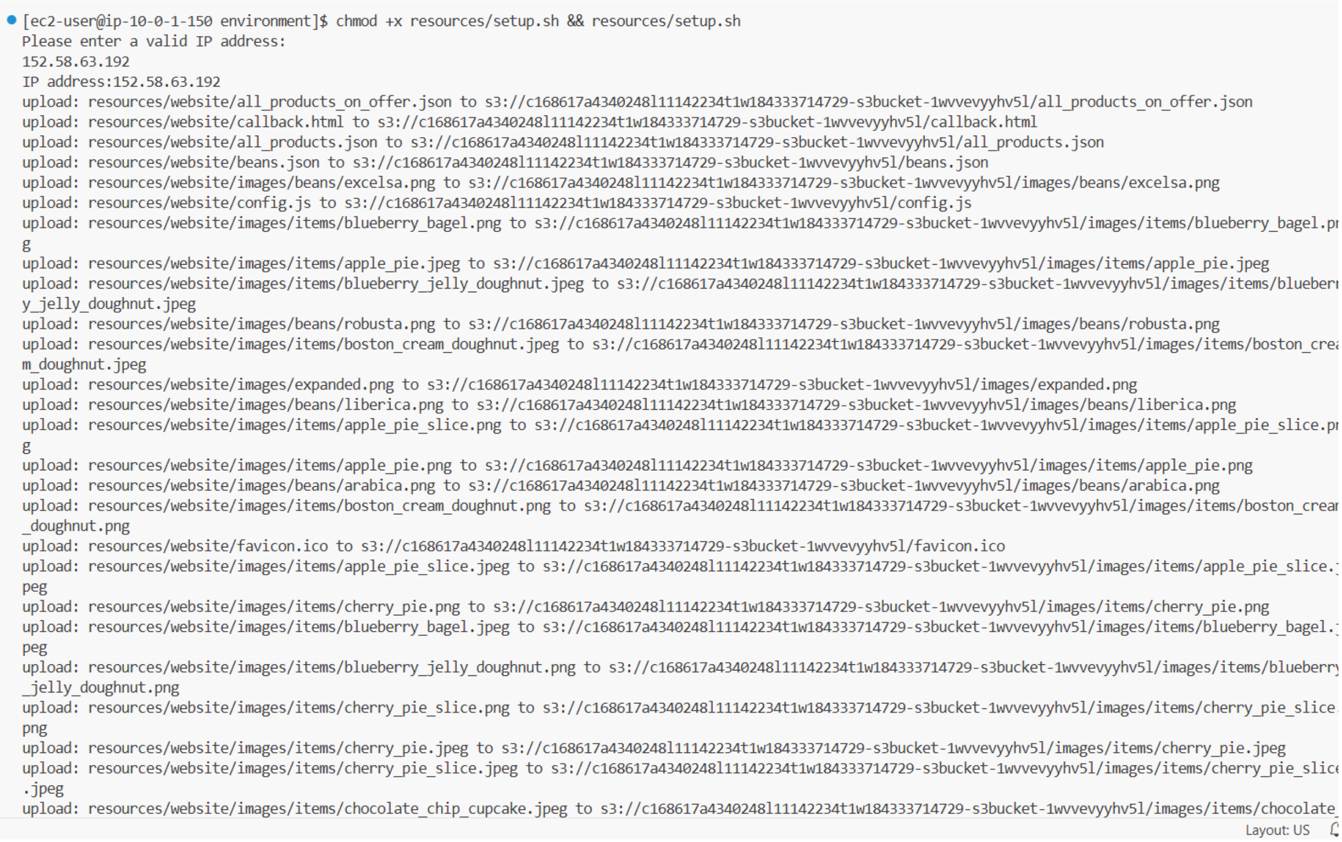


```
2025-08-10 11:15:30 (9.97 MB/s) - '/home/ec2-user/environment/code.zip' saved [5827466/5827466]

[ec2-user@ip-10-0-1-125 environment]$ unzip code.zip
Archive:  code.zip
 extracting: python_3/create_report_code.py
 extracting: python_3/get_all_products_code.py
 extracting: python_3/get_all_products_wrapper.py
 extracting: python_3/create_report_wrapper.py
 extracting: python_3/update_config.py
 extracting: resources/seed.py
 extracting: resources/setup.sh
 extracting: resources/public_policy.json
 extracting: resources/permissions.py
 extracting: resources/website/callback.html
 extracting: resources/website/all_products.json
 extracting: resources/website/all_products_on_offer.json
```

7. Run a script that upgrades the version of the AWS CLI installed on the VS Code IDE.
   - To set permissions on the script and then run it, run the following commands in the Bash terminal:

```
chmod +x ./resources/setup.sh && ./resources/setup.sh
```

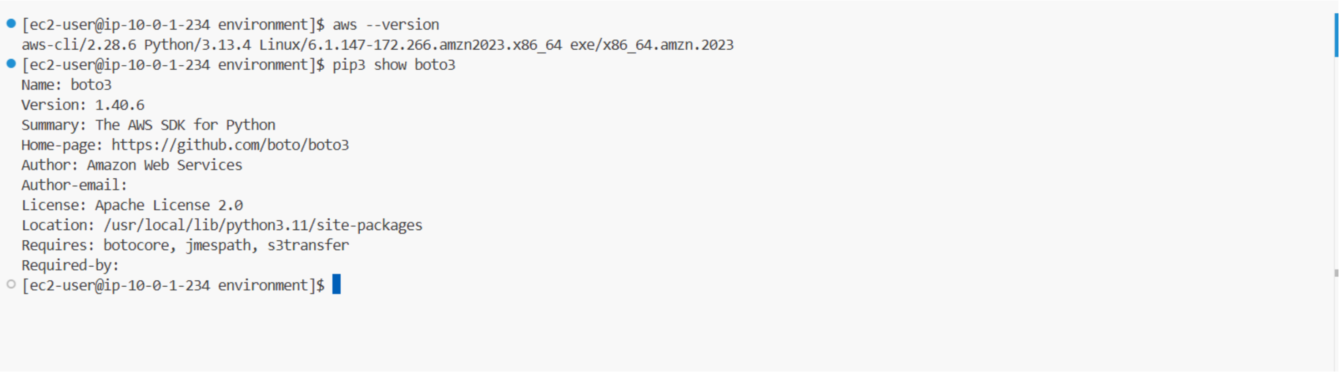
The script will prompt you for the **IP address** by which your computer is known to the internet. Use www.whatismyip.com to discover this address and then paste the IPv4 address into the command prompt and finish running the script.



```
● [ec2-user@ip-10-0-1-150 environment]$ chmod +x resources/setup.sh && resources/setup.sh
  Please enter a valid IP address:
  152.58.63.192
  IP address:152.58.63.192
  upload: resources/website/all_products_on_offer.json to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/all_products_on_offer.json
  upload: resources/website/callback.html to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/callback.html
  upload: resources/website/all_products.json to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/all_products.json
  upload: resources/website/beans.json to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/beans.json
  upload: resources/website/images/beans/excelsa.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/beans/excelsa.png
  upload: resources/website/config.js to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/config.js
  upload: resources/website/images/items/blueberry_bagel.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/blueberry_bagel.pn
  g
  upload: resources/website/images/items/apple_pie.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/apple_pie.jpeg
  upload: resources/website/images/items/blueberry_jelly_doughnut.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/blueberr
  y_jelly_doughnut.jpeg
  upload: resources/website/images/beans/robusta.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/beans/robusta.png
  upload: resources/website/images/items/boston_cream_doughnut.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/boston_crea
  m_doughnut.jpeg
  upload: resources/website/images/expanded.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/expanded.png
  upload: resources/website/images/beans/liberica.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/beans/liberica.png
  upload: resources/website/images/items/apple_pie_slice.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/apple_pie_slice.pn
  g
  upload: resources/website/images/items/apple_pie.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/apple_pie.png
  upload: resources/website/images/beans/arabica.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/beans/arabica.png
  upload: resources/website/images/items/boston_cream_doughnut.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/boston_cream
  _doughnut.png
  upload: resources/website/favicon.ico to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/favicon.ico
  upload: resources/website/images/items/apple_pie_slice.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/apple_pie_slice.
  peg
  upload: resources/website/images/items/cherry_pie.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/cherry_pie.png
  upload: resources/website/images/items/blueberry_bagel.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/blueberry_bagel.
  peg
  upload: resources/website/images/items/blueberry_jelly_doughnut.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/blueberry
  _jelly_doughnut.png
  upload: resources/website/images/items/cherry_pie_slice.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/cherry_pie_slice
  png
  upload: resources/website/images/items/cherry_pie.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/cherry_pie.jpeg
  upload: resources/website/images/items/cherry_pie_slice.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/cherry_pie_slice
  .jpeg
  upload: resources/website/images/items/chocolate_chip_cupcake.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/chocolate_
```

8. Verify the AWS CLI version and also verify that the SDK for Python is installed.
   - Confirm that the AWS CLI is now at version 2 by running the **`aws --version`** command.
   - In the VS Code Bash terminal (at the bottom of the IDE), run the following command:

```
pip3 show boto3
```

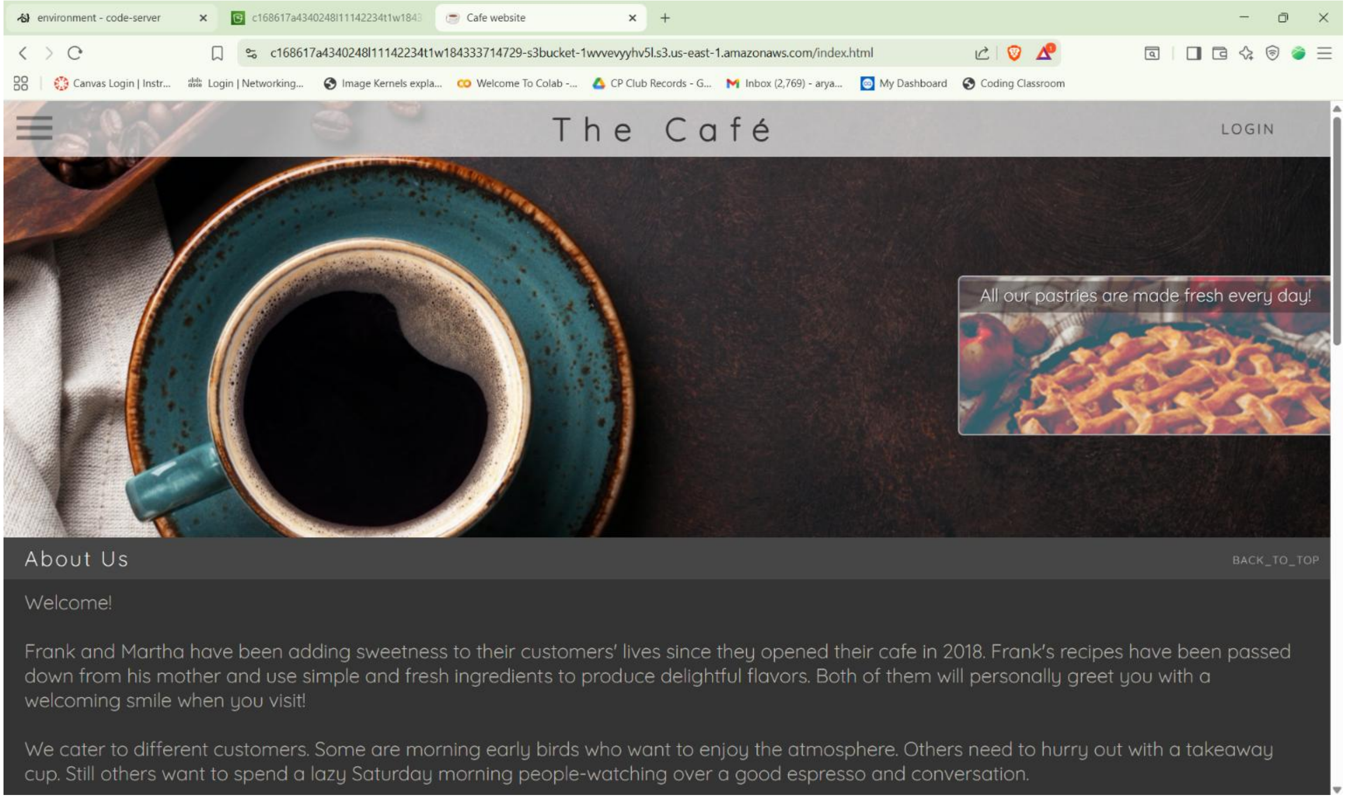


```
[ec2-user@ip-10-0-1-234 environment]$ aws --version
aws-cli/2.28.6 Python/3.13.4 Linux/6.1.147-172.266.amzn2023.x86_64 exe/x86_64.amzn.2023
[ec2-user@ip-10-0-1-234 environment]$ pip3 show boto3
Name: boto3
Version: 1.40.6
Summary: The AWS SDK for Python
Home-page: https://github.com/boto/boto3
Author: Amazon Web Services
Author-email:
License: Apache License 2.0
Location: /usr/local/lib/python3.11/site-packages
Requires: botocore, jmespath, s3transfer
Required-by:
[ec2-user@ip-10-0-1-234 environment]$
```

9. Verify that the cafe website can be loaded in a browser tab.
   - Load the website in a browser tab.
     - In a browser tab, open the Amazon S3 console.
     - Choose your bucket name, and then choose **Objects**.

If the files that the script just uploaded do not display, choose the refresh icon to view them.

- Choose the **index.html** file.
- Copy the **Object URL**. It will be in the following format. https://<bucket-name>.s3.amazon.com/index.html
- Verify that the website displays by pasting the full URL into your browser.



environment - code-server
c168617a4340248l11142234t1w1843
Cafe website
c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l.s3.us-east-1.amazonaws.com/index.html
Canvas Login | Instr...
Login | Networking...
Image Kernels expla...
Welcome To Colab -...
CP Club Records - G...
Inbox (2,769) - arya...
My Dashboard
Coding Classroom
The Café
LOGIN
All our pastries are made fresh every day!
BACK_TO_TOP

About Us

Welcome!

Frank and Martha have been adding sweetness to their customers' lives since they opened their cafe in 2018. Frank's recipes have been passed down from his mother and use simple and fresh ingredients to produce delightful flavors. Both of them will personally greet you with a welcoming smile when you visit!

We cater to different customers. Some are morning early birds who want to enjoy the atmosphere. Others need to hurry out with a takeaway cup. Still others want to spend a lazy Saturday morning people-watching over a good espresso and conversation.

10. Take a moment to see what resources the script created.
    - Confirm that an S3 bucket is hosting the café website files:
        - On the AWS Management Console, in the search bar at the top, search for and select `s3` to open the Amazon S3 console.
        - From the **General purpose buckets**, choose the name of the bucket that was created.
        - Choose **index.html** and copy the **Object URL**.
        - Load the URL in a new browser tab.

          The café website displays. Currently, the website is accessing the hard-coded menu data that is stored in S3 to display the menu information.

          Tip: Notice that several menu items are listed in the **Browse Pastries** section of the page.

    - Confirm that DynamoDB has the menu data stored in a table:
        - On the AWS Management Console, in the search bar at the top, search for and select `DynamoDB` to open the Amazon DynamoDB console.
        - Browse the DynamoDB console.
        - Choose **Tables** and choose the **FoodProducts** table.
        - Choose **Explore table items** and confirm that the table is populated with menu data.
        - Choose **View table details** and then on the **Indexes** tab, confirm that an index named **special_GSI** was created.
    - Confirm that the ProductsApi REST API was defined in API Gateway:
        - On the AWS Management Console, in the search bar at the top, search for and select `API Gateway` to open the Amazon API Gateway console.
        - Browse to the API Gateway console.
        - Choose the name of the **ProductsApi** API.
        - The API has a **GET** method for **/products** and a **GET** method for **/products/on_offer**.
        - Finally, the API has **POST** and **OPTIONS** methods for **/create_report**.
        - From the lower pane, you can use the **TEST** menu and *Test* each method to ensure that they are returning the mock data that you used in the previous lab. Each method should return a 200 HTML status code.
11. Copy the invoke URL for the API to your clipboard.
    - In the API Gateway console, in the left panel, choose **Stages** and then choose the **prod** stage.

      Note: If you see a warning that you do not have ListWebACLs and AssociateWebACL permissions, ignore the warning.

    - Copy the **Invoke URL** value that displays at the top of the page.

      You will use this value in the next step.

12. Update the website's config.js file.

- In the VS Code IDE browser tab, open resources/website/**config.js**.
- On line 2, replace `null` with the invoke URL value that you copied a moment ago. Also, be sure to surround the URL in double quotation marks.
- The file now looks like the following example, but you will have a different value for `<some-value>`:

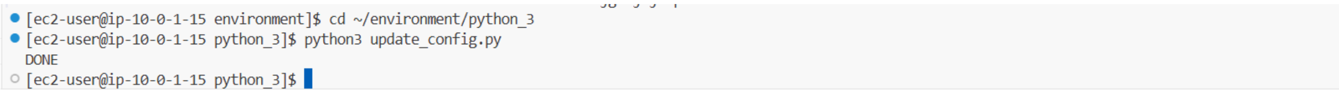
```
window.COFFEE_CONFIG = {
API_GW_BASE_URL_STR: "https://<some-value>.execute-api.us-east-
1.amazonaws.com/prod",
COGNITO_LOGIN_BASE_URL_STR: null
};
```

- Verify that `/prod` appears at the end of the URL with no trailing slash.
- Close the file by choosing **X** from the top. (Your changes are saved automatically).

13. Update and then run the update_config.py script.

- Open python_3/**update_config.py** in the text editor.
- Replace the <FMI_1> placeholder with the name of your S3 bucket.
- **Tip**: Find the bucket name in the S3 console, or run the following command:
  `aws s3 ls`
- Notice that this script will upload the config.js file that you just edited to the S3 bucket.
- Close the file by choosing **X** from the top. (Your changes are saved automatically).
- To run the script, run the following commands:

```
cd ~/environment/python_3
python3 update_config.py
```



```
● [ec2-user@ip-10-0-1-15 environment]$ cd ~/environment/python_3
● [ec2-user@ip-10-0-1-15 python_3]$ python3 update_config.py
  DONE
○ [ec2-user@ip-10-0-1-15 python_3]$ 
```

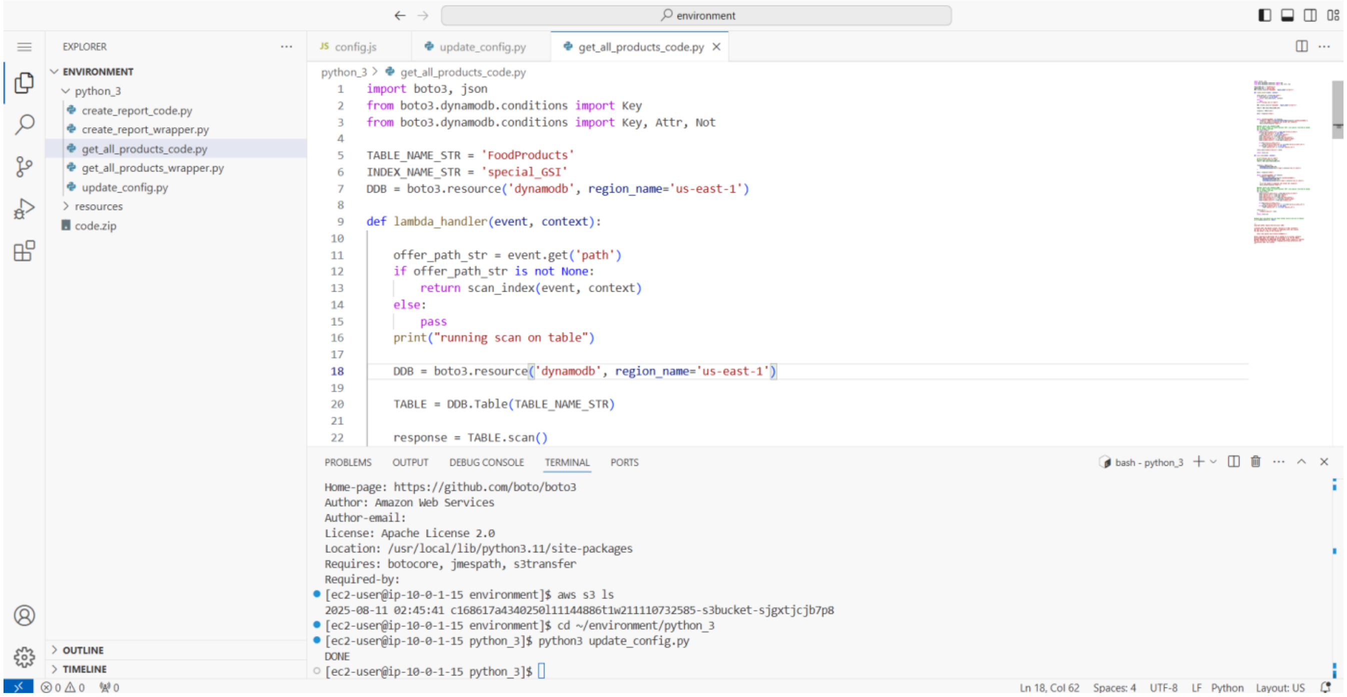
Task 2: Creating a Lambda function to retrieve data from DynamoDB

14. Observe and edit the Python code that you will use in the Lambda function.
    - In the VS Code IDE file browser, browse to and open python_3/**get_all_products_code.py**.
    - Replace the <FMI_1> placeholder and the <FMI_2> placeholder with the proper values.

Tip: To find the missing values in the code, return to the DynamoDB console.

- Notice that the code does the following:
  - It creates a boto3 client to interact with the DynamoDB service.
  - It reads the items out of the table and returns the menu data.

- It also scans the table index and filters for items that are in stock.
- Close the file by choosing **X** from the top. (Your changes are saved automatically).

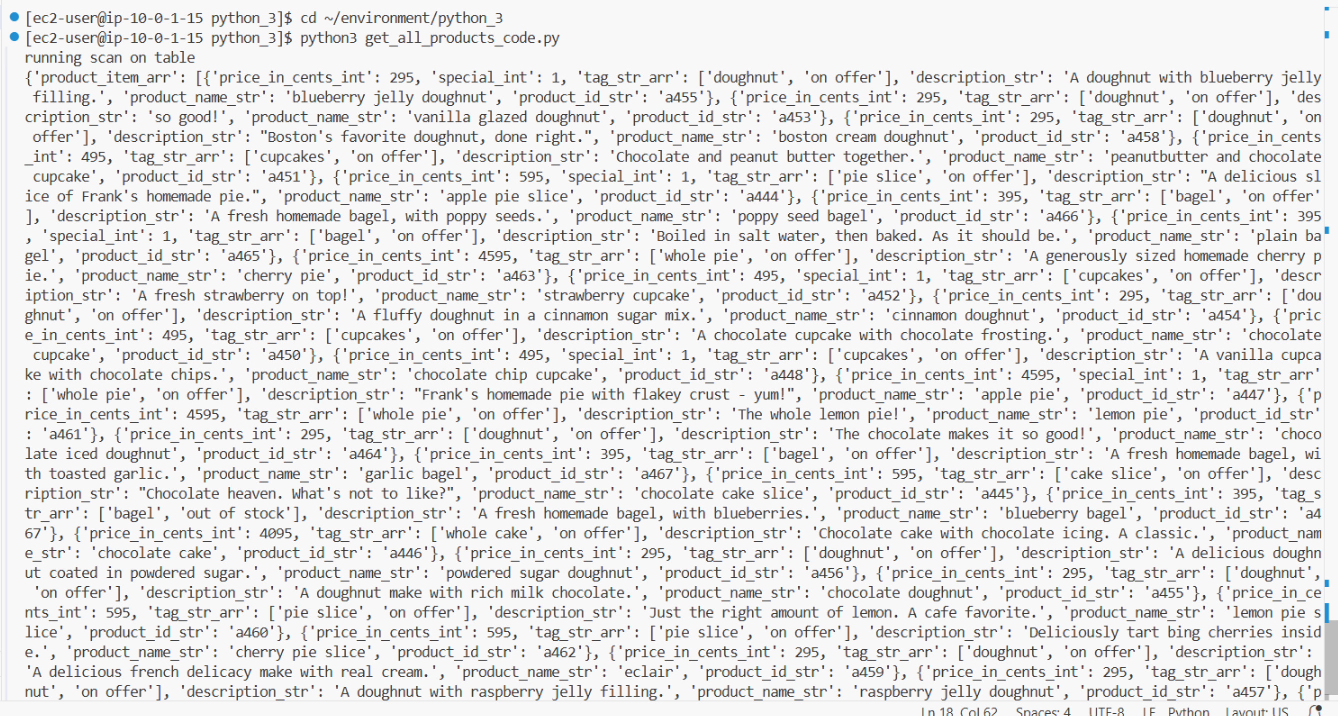


```python
import boto3, json
from boto3.dynamodb.conditions import Key
from boto3.dynamodb.conditions import Key, Attr, Not

TABLE_NAME_STR = 'FoodProducts'
INDEX_NAME_STR = 'special_GSI'
DDB = boto3.resource('dynamodb', region_name='us-east-1')

def lambda_handler(event, context):

    offer_path_str = event.get('path')
    if offer_path_str is not None:
        return scan_index(event, context)
    else:
        pass
    print("running scan on table")

    DDB = boto3.resource('dynamodb', region_name='us-east-1')

    TABLE = DDB.Table(TABLE_NAME_STR)

    response = TABLE.scan()
```

```
Home-page: https://github.com/boto/boto3
Author: Amazon Web Services
Author-email:
License: Apache License 2.0
Location: /usr/local/lib/python3.11/site-packages
Requires: botocore, jmespath, s3transfer
Required-by:
[ec2-user@ip-10-0-1-15 environment]$ aws s3 ls
2025-08-11 02:45:41 c168617a4340250l11144886t1w211110732585-s3bucket-sjgxtjcjb7p8
[ec2-user@ip-10-0-1-15 environment]$ cd ~/environment/python_3
[ec2-user@ip-10-0-1-15 python_3]$ python3 update_config.py
DONE
[ec2-user@ip-10-0-1-15 python_3]$
```

15. Test the code *locally* in VS Code IDE.

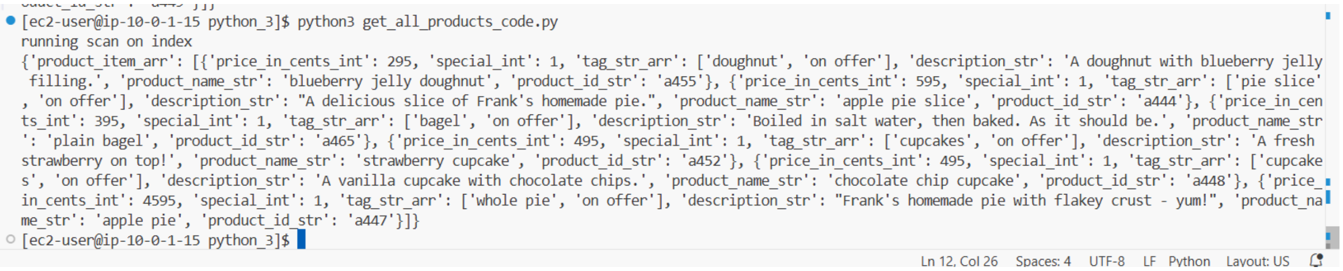
- To ensure that you are in the correct folder, run the following command:
  `cd ~/environment/python_3`
- To run the code locally in the VS Code IDE terminal, run the following command:
  `python3 get_all_products_code.py`



```
[ec2-user@ip-10-0-1-15 python_3]$ cd ~/environment/python_3
[ec2-user@ip-10-0-1-15 python_3]$ python3 get_all_products_code.py
running scan on table
{'product_item_arr': [{'price_in_cents_int': 295, 'special_int': 1, 'tag_str_arr': ['doughnut', 'on offer'], 'description_str': 'A doughnut with blueberry jelly
 filling.', 'product_name_str': 'blueberry jelly doughnut', 'product_id_str': 'a455'}, {'price_in_cents_int': 295, 'tag_str_arr': ['doughnut', 'on offer'], 'des
cription_str': 'so good!', 'product_name_str': 'vanilla glazed doughnut', 'product_id_str': 'a453'}, {'price_in_cents_int': 295, 'tag_str_arr': ['doughnut', 'on
 offer'], 'description_str': "Boston's favorite doughnut, done right.", 'product_name_str': 'boston cream doughnut', 'product_id_str': 'a458'}, {'price_in_cents
_int': 495, 'tag_str_arr': ['cupcakes', 'on offer'], 'description_str': 'Chocolate and peanut butter together.', 'product_name_str': 'peanutbutter and chocolate
 cupcake', 'product_id_str': 'a451'}, {'price_in_cents_int': 595, 'special_int': 1, 'tag_str_arr': ['pie slice', 'on offer'], 'description_str': "A delicious sl
ice of Frank's homemade pie.", 'product_name_str': 'apple pie slice', 'product_id_str': 'a444'}, {'price_in_cents_int': 395, 'tag_str_arr': ['bagel', 'on offer'
], 'description_str': 'A fresh homemade bagel, with poppy seeds.', 'product_name_str': 'poppy seed bagel', 'product_id_str': 'a466'}, {'price_in_cents_int': 395
, 'special_int': 1, 'tag_str_arr': ['bagel', 'on offer'], 'description_str': 'Boiled in salt water, then baked. As it should be.', 'product_name_str': 'plain ba
gel', 'product_id_str': 'a465'}, {'price_in_cents_int': 4595, 'tag_str_arr': ['whole pie', 'on offer'], 'description_str': 'A generously sized homemade cherry p
ie.', 'product_name_str': 'cherry pie', 'product_id_str': 'a463'}, {'price_in_cents_int': 495, 'special_int': 1, 'tag_str_arr': ['cupcakes', 'on offer'], 'descr
iption_str': 'A fresh strawberry on top!', 'product_name_str': 'strawberry cupcake', 'product_id_str': 'a452'}, {'price_in_cents_int': 295, 'tag_str_arr': ['dou
ghnut', 'on offer'], 'description_str': 'A fluffy doughnut in a cinnamon sugar mix.', 'product_name_str': 'cinnamon doughnut', 'product_id_str': 'a454'}, {'pric
e_in_cents_int': 495, 'tag_str_arr': ['cupcakes', 'on offer'], 'description_str': 'A chocolate cupcake with chocolate frosting.', 'product_name_str': 'chocolate
 cupcake', 'product_id_str': 'a450'}, {'price_in_cents_int': 495, 'special_int': 1, 'tag_str_arr': ['cupcakes', 'on offer'], 'description_str': 'A vanilla cupca
ke with chocolate chips.', 'product_name_str': 'chocolate chip cupcake', 'product_id_str': 'a448'}, {'price_in_cents_int': 4595, 'special_int': 1, 'tag_str_arr'
: ['whole pie', 'on offer'], 'description_str': "Frank's homemade pie with flakey crust - yum!", 'product_name_str': 'apple pie', 'product_id_str': 'a447'}, {'p
rice_in_cents_int': 4595, 'tag_str_arr': ['whole pie', 'on offer'], 'description_str': 'The whole lemon pie!', 'product_name_str': 'lemon pie', 'product_id_str'
: 'a461'}, {'price_in_cents_int': 295, 'tag_str_arr': ['doughnut', 'on offer'], 'description_str': 'The chocolate makes it so good!', 'product_name_str': 'choco
late iced doughnut', 'product_id_str': 'a464'}, {'price_in_cents_int': 395, 'tag_str_arr': ['bagel', 'on offer'], 'description_str': 'A fresh homemade bagel, wi
th toasted garlic.', 'product_name_str': 'garlic bagel', 'product_id_str': 'a467'}, {'price_in_cents_int': 595, 'tag_str_arr': ['cake slice', 'on offer'], 'desc
ription_str': "Chocolate heaven. What's not to like?", 'product_name_str': 'chocolate cake slice', 'product_id_str': 'a445'}, {'price_in_cents_int': 395, 'tag_s
tr_arr': ['bagel', 'out of stock'], 'description_str': 'A fresh homemade bagel, with blueberries.', 'product_name_str': 'blueberry bagel', 'product_id_str': 'a4
67'}, {'price_in_cents_int': 4095, 'tag_str_arr': ['whole cake', 'on offer'], 'description_str': 'Chocolate cake with chocolate icing. A classic.', 'product_nam
e_str': 'chocolate cake', 'product_id_str': 'a446'}, {'price_in_cents_int': 295, 'tag_str_arr': ['doughnut', 'on offer'], 'description_str': 'A delicious doughn
ut coated in powdered sugar.', 'product_name_str': 'powdered sugar doughnut', 'product_id_str': 'a456'}, {'price_in_cents_int': 295, 'tag_str_arr': ['doughnut',
 'on offer'], 'description_str': 'A doughnut make with rich milk chocolate.', 'product_name_str': 'chocolate doughnut', 'product_id_str': 'a455'}, {'price_in_ce
nts_int': 595, 'tag_str_arr': ['pie slice', 'on offer'], 'description_str': 'Just the right amount of lemon. A cafe favorite.', 'product_name_str': 'lemon pie s
lice', 'product_id_str': 'a460'}, {'price_in_cents_int': 595, 'tag_str_arr': ['pie slice', 'on offer'], 'description_str': 'Deliciously tart bing cherries insid
e.', 'product_name_str': 'cherry pie slice', 'product_id_str': 'a462'}, {'price_in_cents_int': 295, 'tag_str_arr': ['doughnut', 'on offer'], 'description_str':
'A delicious french delicacy make with real cream.', 'product_name_str': 'eclair', 'product_id_str': 'a459'}, {'price_in_cents_int': 295, 'tag_str_arr': ['dough
nut', 'on offer'], 'description_str': 'A doughnut with raspberry jelly filling.', 'product_name_str': 'raspberry jelly doughnut', 'product_id_str': 'a457'}, {'p
```

16. Modify a setting in the code and test it again.

- In the **get_all_products_code.py** file, line 12 has the following code:
  `if offer_path_str is not None:`
- **Analysis**: If the offer_path_str variable is not found, the condition fails and runs a scan of the table.
- To verify that this logic is working, temporarily reverse this condition. Remove the word not from this line of code, so that it looks like the following:
  `if offer_path_str is None:`
- Close the file by choosing **X** from the top. (Your changes are saved automatically).
- Run the code again:
  `python3 get_all_products_code.py`



```
[ec2-user@ip-10-0-1-15 python_3]$ python3 get_all_products_code.py
running scan on index
{'product_item_arr': [{'price_in_cents_int': 295, 'special_int': 1, 'tag_str_arr': ['doughnut', 'on offer'], 'description_str': 'A doughnut with blueberry jelly
 filling.', 'product_name_str': 'blueberry jelly doughnut', 'product_id_str': 'a455'}, {'price_in_cents_int': 595, 'special_int': 1, 'tag_str_arr': ['pie slice'
, 'on offer'], 'description_str': "A delicious slice of Frank's homemade pie.", 'product_name_str': 'apple pie slice', 'product_id_str': 'a444'}, {'price_in_cen
ts_int': 395, 'special_int': 1, 'tag_str_arr': ['bagel', 'on offer'], 'description_str': 'Boiled in salt water, then baked. As it should be.', 'product_name_str
': 'plain bagel', 'product_id_str': 'a465'}, {'price_in_cents_int': 495, 'special_int': 1, 'tag_str_arr': ['cupcakes', 'on offer'], 'description_str': 'A fresh
strawberry on top!', 'product_name_str': 'strawberry cupcake', 'product_id_str': 'a452'}, {'price_in_cents_int': 495, 'special_int': 1, 'tag_str_arr': ['cupcake
s', 'on offer'], 'description_str': 'A vanilla cupcake with chocolate chips.', 'product_name_str': 'chocolate chip cupcake', 'product_id_str': 'a448'}, {'price_
in_cents_int': 4595, 'special_int': 1, 'tag_str_arr': ['whole pie', 'on offer'], 'description_str': "Frank's homemade pie with flakey crust - yum!", 'product_na
me_str': 'apple pie', 'product_id_str': 'a447'}]}
[ec2-user@ip-10-0-1-15 python_3]$
```

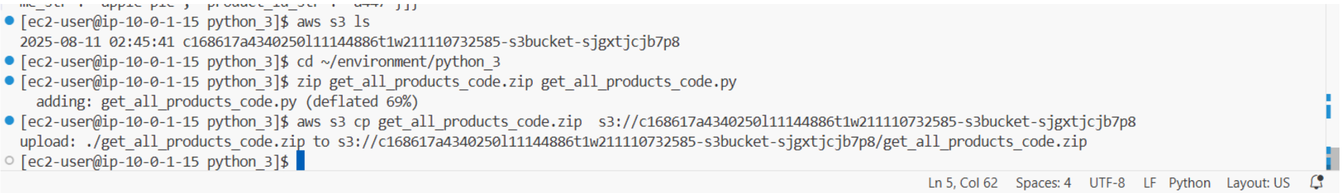
17. Package the code and store it in an S3 bucket.
    - A bucket with -s3bucket- in the name was created for you when you started the lab.
    - Verify that your VS Code IDE terminal is in the **python_3** directory.

      ```
      cd ~/environment/python_3
      ```
    - To place a copy of your code in a .zip file, run the following command:

      ```
      zip get_all_products_code.zip get_all_products_code.py
      ```
    - Next, to retrieve the name of your S3 bucket, run the following command:

      ```
      aws s3 ls
      ```
    - Finally, to place the .zip file in the bucket, run the following command. Replace <bucket-name> with the actual bucket name that you retrieved:

      ```
      aws s3 cp get_all_products_code.zip  s3://<bucket-name>
      ```

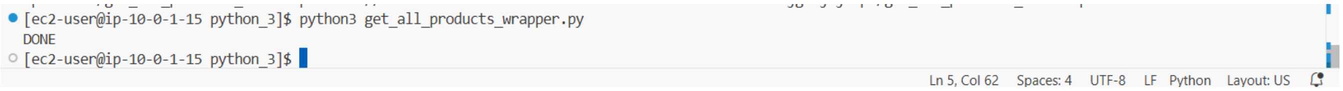


```
● [ec2-user@ip-10-0-1-15 python_3]$ aws s3 ls
  2025-08-11 02:45:41 c168617a4340250l111144886t1w211110732585-s3bucket-sjgxtjcjb7p8
● [ec2-user@ip-10-0-1-15 python_3]$ cd ~/environment/python_3
● [ec2-user@ip-10-0-1-15 python_3]$ zip get_all_products_code.zip get_all_products_code.py
    adding: get_all_products_code.py (deflated 69%)
● [ec2-user@ip-10-0-1-15 python_3]$ aws s3 cp get_all_products_code.zip  s3://c168617a4340250l111144886t1w211110732585-s3bucket-sjgxtjcjb7p8
  upload: ./get_all_products_code.zip to s3://c168617a4340250l111144886t1w211110732585-s3bucket-sjgxtjcjb7p8/get_all_products_code.zip
○ [ec2-user@ip-10-0-1-15 python_3]$
```

18. To create the Lambda function, run the following command:

    ```
    python3 get_all_products_wrapper.py
    ```

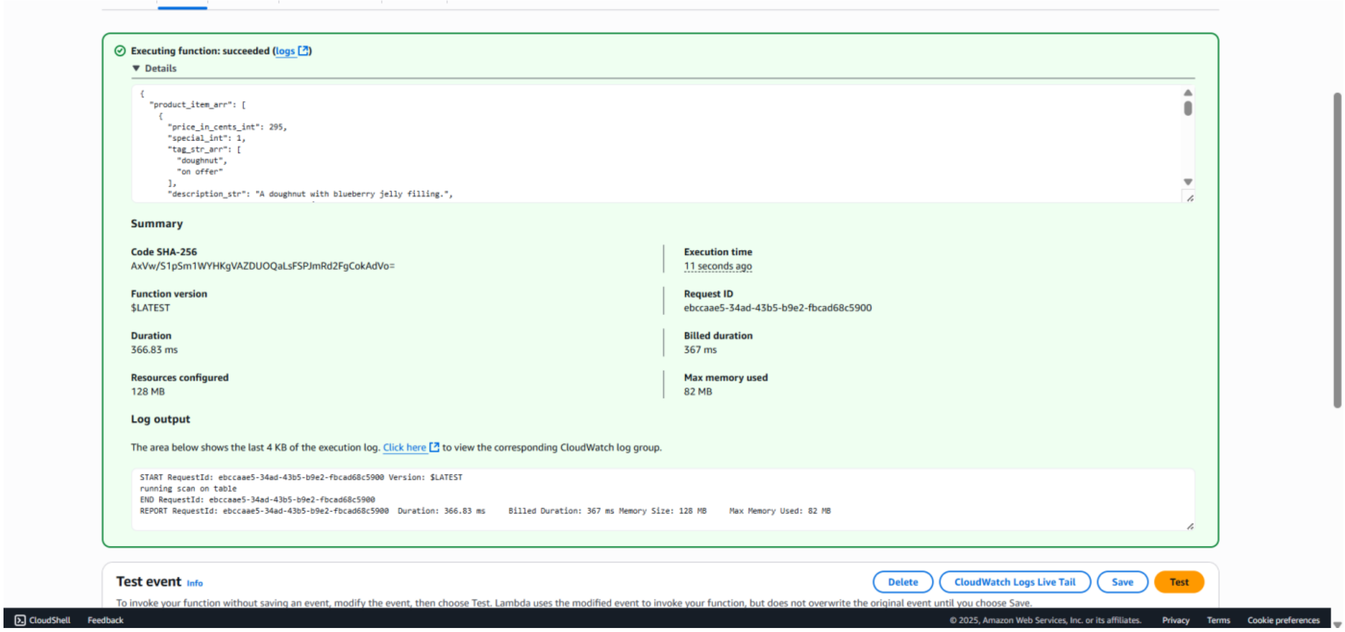
- The output of the command shows DONE, confirming that the code ran without errors.



```
● [ec2-user@ip-10-0-1-15 python_3]$ python3 get_all_products_wrapper.py
  DONE
○ [ec2-user@ip-10-0-1-15 python_3]$
```

19. Observe the function that you created and test it.

- Browse to the Lambda console.
- Choose the name of the **get_all_products** function that you just created.
- In the **Code source** panel, open (double-click) the **get_all_products_code.py** file to display the code.
- Choose **Test**.
- For **Event name**, enter Products
- Keep all of the other default test event values, and choose **Save**.
    - The test event is saved.
- Choose **Test** again.



Executing function: succeeded (logs)

Details

```
{
  "product_item_arr": [
    {
      "price_in_cents_int": 295,
      "special_int": 1,
      "tag_str_arr": [
        "doughnut",
        "on offer"
      ],
      "description_str": "A doughnut with blueberry jelly filling.",
```

Summary

Code SHA-256
AxVw/S1pSm1WYHKgVAZDUOQaLsFSPJmRd2FgCokAdVo=

Execution time
11 seconds ago

Function version
$LATEST

Request ID
ebccaae5-34ad-43b5-b9e2-fbcad68c5900

Duration
366.83 ms

Billed duration
367 ms

Resources configured
128 MB

Max memory used
82 MB

Log output

The area below shows the last 4 KB of the execution log. Click here to view the corresponding CloudWatch log group.

```
START RequestId: ebccaae5-34ad-43b5-b9e2-fbcad68c5900 Version: $LATEST
running scan on table
END RequestId: ebccaae5-34ad-43b5-b9e2-fbcad68c5900
REPORT RequestId: ebccaae5-34ad-43b5-b9e2-fbcad68c5900  Duration: 366.83 ms     Billed Duration: 367 ms Memory Size: 128 MB     Max Memory Used: 82 MB
```

Test event Info

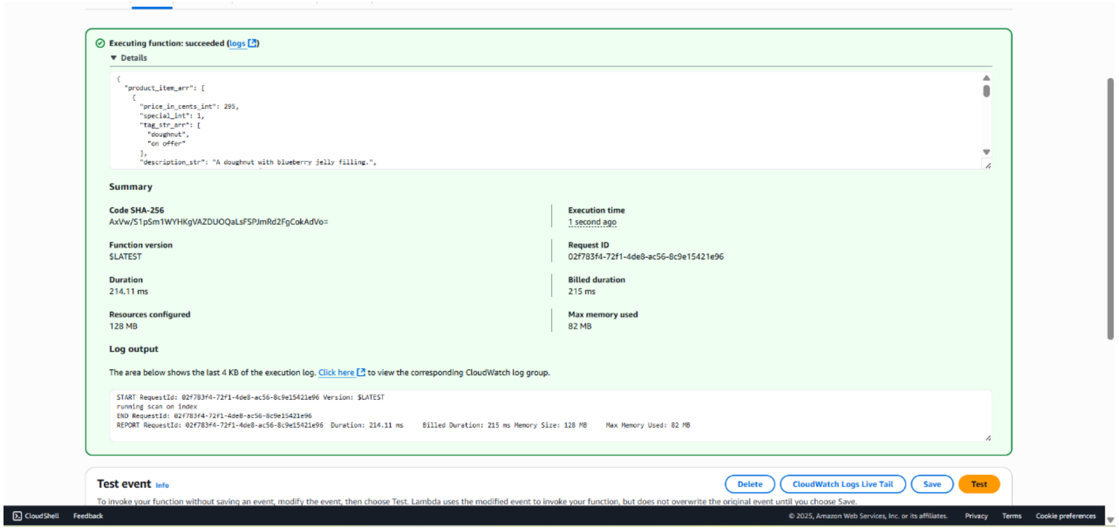
To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save.

Delete
CloudWatch Logs Live Tail
Save
Test
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

20. Create a new test event that is called **onOffer**.

- In the **Code source** panel, open the **Test** menu (choose the arrow icon), and choose **Configure test event**.

- Choose **Create new event**.
  - For **Event name**, enter onOffer
  - In the code editor panel, replace the existing code with the following:



Executing function: succeeded (logs)

Details

```
{
  "product_item_arr": [
    {
      "price_in_cents_int": 295,
      "special_int": 1,
      "tag_str_arr": [
        "doughnut",
        "on offer"
      ],
      "description_str": "A doughnut with blueberry jelly filling.",
```

Summary

Code SHA-256
AxVw/S1pSm1WYHKgVAZDUOQaLsFSPJmRd2FgCokAdVo=

Execution time
1 second ago

Function version
$LATEST

Request ID
02f783f4-72f1-4de8-ac56-8c9e15421e96

Duration
214.11 ms

Billed duration
215 ms

Resources configured
128 MB

Max memory used
82 MB

Log output

The area below shows the last 4 KB of the execution log. Click here to view the corresponding CloudWatch log group.

```
START RequestId: 02f783f4-72f1-4de8-ac56-8c9e15421e96 Version: $LATEST
running scan on index
END RequestId: 02f783f4-72f1-4de8-ac56-8c9e15421e96
REPORT RequestId: 02f783f4-72f1-4de8-ac56-8c9e15421e96  Duration: 214.11 ms     Billed Duration: 215 ms Memory Size: 128 MB     Max Memory Used: 82 MB
```

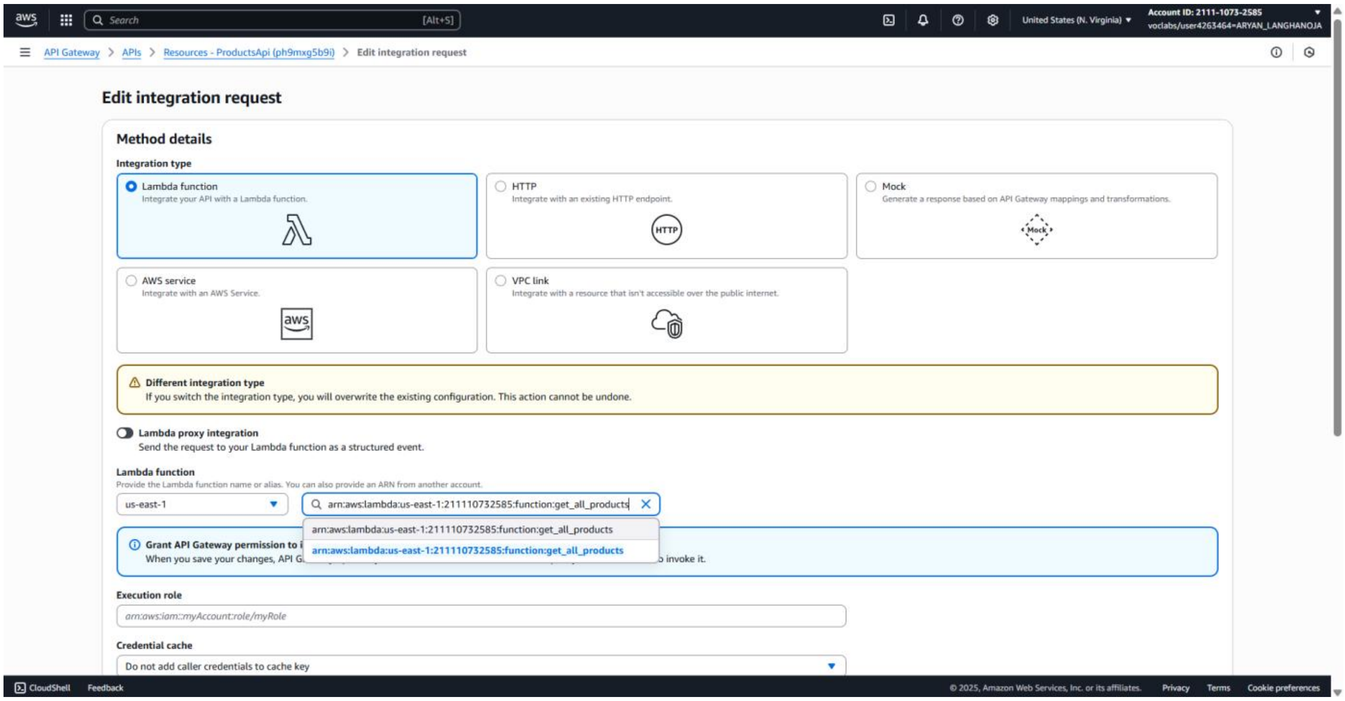
Test event Info

To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save.

Delete
CloudWatch Logs Live Tail
Save
Test
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Task 3: Configuring the REST API to invoke the Lambda function

21. Replace the mock endpoint with the Lambda function.
    - At the top of the page. Ensure that the **GET** method is still selected under **/products**.
    - Choose **Integration Request** and **Edit**:
    - Integration type: **Lambda Function**
    - Lambda Region: **us-east-1**
    - Lambda Function: get_all_products
    - Choose **Save**
    - Choose **Save**.
    - Notice on the right side of the page that the method is no longer calling a "Mock Endpoint". Instead, it is calling your Lambda function.



aws
Search [Alt+S]
United States (N. Virginia) ▾
Account ID: 2111-1073-2585
voclabs/user4263464=ARYAN_LANGHANOJA
API Gateway > APIs > Resources - ProductsApi (ph9mxg5b9i) > Edit integration request
HTTP
Mock
aws
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Edit integration request

Method details

Integration type

● Lambda function
Integrate your API with a Lambda function.

○ HTTP
Integrate with an existing HTTP endpoint.

○ Mock
Generate a response based on API Gateway mappings and transformations.

○ AWS service
Integrate with an AWS Service.

○ VPC link
Integrate with a resource that isn't accessible over the public internet.

⚠ Different integration type
If you switch the integration type, you will overwrite the existing configuration. This action cannot be undone.

Lambda proxy integration
Send the request to your Lambda function as a structured event.

Lambda function
Provide the Lambda function name or alias. You can also provide an ARN from another account.

us-east-1 ▾

arn:aws:lambda:us-east-1:211110732585:function:get_all_products ✕

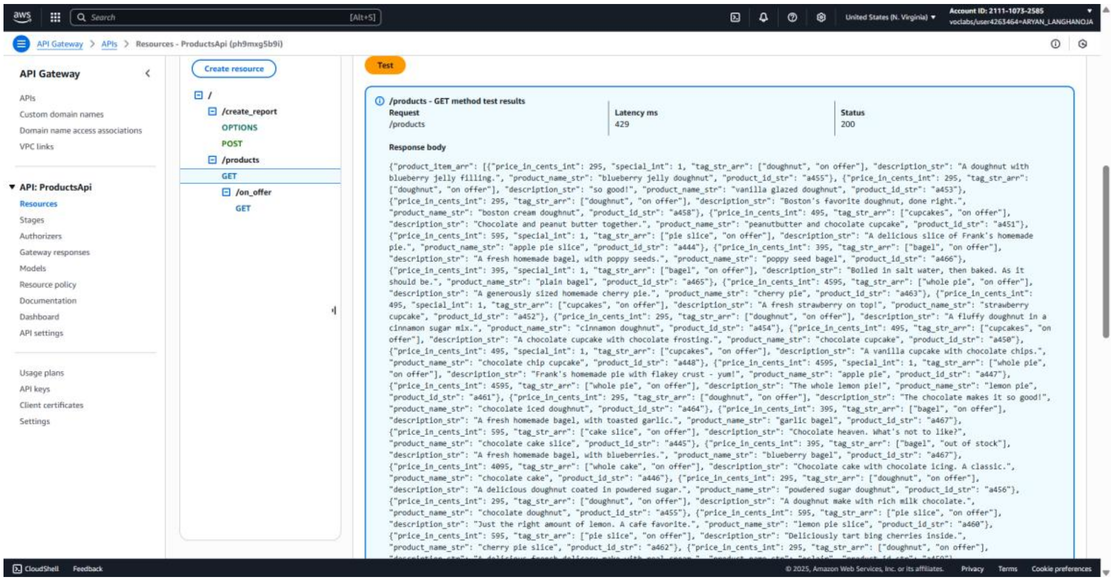
arn:aws:lambda:us-east-1:211110732585:function:get_all_products

arn:aws:lambda:us-east-1:211110732585:function:get_all_products

ⓘ Grant API Gateway permission to i
When you save your changes, API G
o invoke it.

Execution role
arn:aws:iam::myAccount:role/myRole

Credential cache
Do not add caller credentials to cache key ▾



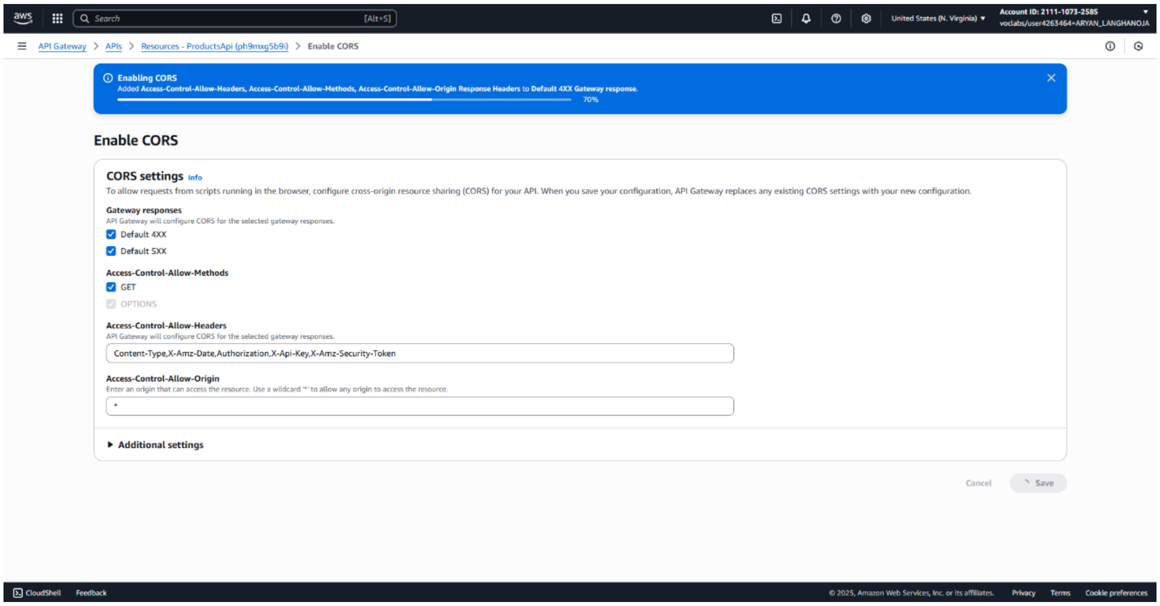
/products - GET method test results

Request: /products
Latency ms: 429
Status: 200

Response body

```
{"product_item_arr": [{"price_in_cents_int": 295, "special_int": 1, "tag_str_arr": ["doughnut", "on offer"], "description_str": "A doughnut with
blueberry jelly filling.", "product_name_str": "blueberry jelly doughnut", "product_id_str": "a455"}, {"price_in_cents_int": 295, "tag_str_arr":
["doughnut", "on offer"], "description_str": "so good!", "product_name_str": "vanilla glazed doughnut", "product_id_str": "a453"},
{"price_in_cents_int": 295, "tag_str_arr": ["doughnut", "on offer"], "description_str": "Boston's favorite doughnut, done right.",
"product_name_str": "boston cream doughnut", "product_id_str": "a458"}, {"price_in_cents_int": 495, "tag_str_arr": ["cupcakes", "on offer"],
"description_str": "Chocolate and peanut butter together.", "product_name_str": "peanutbutter and chocolate cupcake", "product_id_str": "a451"},
{"price_in_cents_int": 595, "special_int": 1, "tag_str_arr": ["pie slice", "on offer"], "description_str": "A delicious slice of Frank's homemade
pie.", "product_name_str": "apple pie slice", "product_id_str": "a444"}, {"price_in_cents_int": 395, "tag_str_arr": ["bagel", "on offer"],
"description_str": "A fresh homemade bagel, with poppy seeds.", "product_name_str": "poppy seed bagel", "product_id_str": "a466"},
{"price_in_cents_int": 395, "special_int": 1, "tag_str_arr": ["bagel", "on offer"], "description_str": "Boiled in salt water, then baked. As it
should be.", "product_name_str": "plain bagel", "product_id_str": "a465"}, {"price_in_cents_int": 4595, "tag_str_arr": ["whole pie", "on offer"],
"description_str": "A generously sized homemade cherry pie.", "product_name_str": "cherry pie", "product_id_str": "a463"}, {"price_in_cents_int":
495, "special_int": 1, "tag_str_arr": ["cupcakes", "on offer"], "description_str": "A fresh strawberry on top!", "product_name_str": "strawberry
cupcake", "product_id_str": "a452"}, {"price_in_cents_int": 295, "tag_str_arr": ["doughnut", "on offer"], "description_str": "A fluffy doughnut in a
cinnamon sugar mix.", "product_name_str": "cinnamon doughnut", "product_id_str": "a454"}, {"price_in_cents_int": 495, "tag_str_arr": ["cupcakes", "on
offer"], "description_str": "A chocolate cupcake with chocolate frosting.", "product_name_str": "chocolate cupcake", "product_id_str": "a450"},
{"price_in_cents_int": 495, "special_int": 1, "tag_str_arr": ["cupcakes", "on offer"], "description_str": "A vanilla cupcake with chocolate chips.",
"product_name_str": "chocolate chip cupcake", "product_id_str": "a448"}, {"price_in_cents_int": 4595, "special_int": 1, "tag_str_arr": ["whole pie",
"on offer"], "description_str": "Frank's homemade pie with flakey crust - yum!", "product_name_str": "apple pie", "product_id_str": "a447"},
{"price_in_cents_int": 4595, "tag_str_arr": ["whole pie", "on offer"], "description_str": "The whole lemon pie!", "product_name_str": "lemon pie",
"product_id_str": "a461"}, {"price_in_cents_int": 295, "tag_str_arr": ["doughnut", "on offer"], "description_str": "The chocolate makes it so good!",
"product_name_str": "chocolate iced doughnut", "product_id_str": "a464"}, {"price_in_cents_int": 395, "tag_str_arr": ["bagel", "on offer"],
"description_str": "A fresh homemade bagel, with toasted garlic.", "product_name_str": "garlic bagel", "product_id_str": "a467"},
{"price_in_cents_int": 595, "tag_str_arr": ["cake slice", "on offer"], "description_str": "Chocolate heaven. What's not to like?",
"product_name_str": "chocolate cake slice", "product_id_str": "a445"}, {"price_in_cents_int": 395, "tag_str_arr": ["bagel", "out of stock"],
"description_str": "A fresh homemade bagel, with blueberries.", "product_name_str": "blueberry bagel", "product_id_str": "a467"},
{"price_in_cents_int": 4095, "tag_str_arr": ["whole cake", "on offer"], "description_str": "Chocolate cake with chocolate icing. A classic.",
"product_name_str": "chocolate cake", "product_id_str": "a446"}, {"price_in_cents_int": 295, "tag_str_arr": ["doughnut", "on offer"],
"description_str": "A delicious doughnut coated in powdered sugar.", "product_name_str": "powdered sugar doughnut", "product_id_str": "a456"},
{"price_in_cents_int": 295, "tag_str_arr": ["doughnut", "on offer"], "description_str": "A doughnut make with rich milk chocolate.",
"product_name_str": "chocolate doughnut", "product_id_str": "a455"}, {"price_in_cents_int": 595, "tag_str_arr": ["pie slice", "on offer"],
"description_str": "Just the right amount of lemon. A cafe favorite.", "product_name_str": "lemon pie slice", "product_id_str": "a460"},
{"price_in_cents_int": 595, "tag_str_arr": ["pie slice", "on offer"], "description_str": "Deliciously tart bing cherries inside.",
"product_name_str": "cherry pie slice", "product_id_str": "a462"}, {"price_in_cents_int": 295, "tag_str_arr": ["doughnut", "on offer"],
```

30. Re-enable CORS on the /products API resource.
    - Choose **/products** so that it is highlighted.
    - Select the **GET** method.
    - Choose button **Enable CORS** from the top.
    - Select **Default 4XX** and **Default 5XX** under **Gateway responses**.
    - Select **GET** under **Access-Control-Allow-Methods**.
    - Choose **Save**.



Enabling CORS
Added Access-Control-Allow-Headers, Access-Control-Allow-Methods, Access-Control-Allow-Origin Response Headers to Default 4XX Gateway response.
70%

Enable CORS

CORS settings

To allow requests from scripts running in the browser, configure cross-origin resource sharing (CORS) for your API. When you save your configuration, API Gateway replaces any existing CORS settings with your new configuration.

Gateway responses
API Gateway will configure CORS for the selected gateway responses.
☑ Default 4XX
☑ Default 5XX

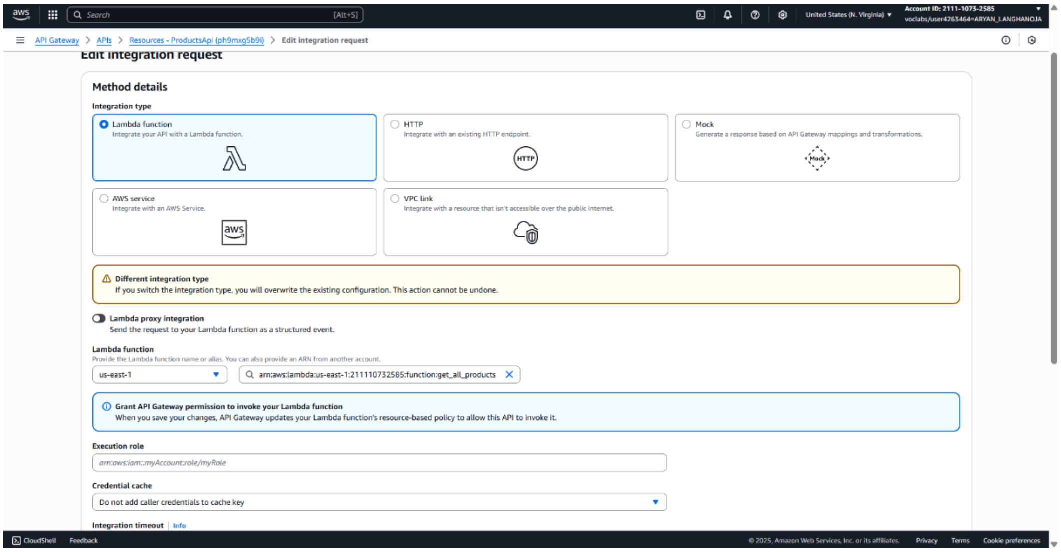
Access-Control-Allow-Methods
☑ GET
☑ OPTIONS

Access-Control-Allow-Headers
API Gateway will configure CORS for the selected gateway responses.
Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token

Access-Control-Allow-Origin
Enter an origin that can access the resource. Use a wildcard '*' to allow any origin to access the resource.
*

API Gateway > APIs > Resources - ProductsApi (ph9mxg5b9i) > Enable CORS
Info
Additional settings
Cancel
Save
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

31. Using the same approach, update the /on_offer GET API method.
   - Choose the **ProductsApi** API, and choose the **GET** method for **/on_offer**.
   - Choose **Integration Request** and configure:
     1. Integration type: **Lambda Function**
     2. Lambda Region: **us-east-1**
     3. Lambda Function: `get_all_products`
   - Choose **Save**.
   - Choose **/on_offer** so that it is highlighted.
   - Choose **Enable CORS**.
   - Select **Default 4XX** and **Default 5XX** under **Gateway responses**.
   - Select **GET** under **Access-Control-Allow-Methods**.
   - Choose **Save**.


aws
Search
[Alt+S]
United States (N. Virginia) ▾
Account ID: 2111-1073-2585
voclabs/user4263464=ARYAN_LANGHANOJA
API Gateway > APIs > Resources - ProductsApi (ph9mxg5b9i) > Edit integration request
HTTP
Mock
aws
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Edit integration request

Method details

Integration type

● Lambda function
Integrate your API with a Lambda function.

○ HTTP
Integrate with an existing HTTP endpoint.

○ Mock
Generate a response based on API Gateway mappings and transformations.

○ AWS service
Integrate with an AWS Service.

○ VPC link
Integrate with a resource that isn't accessible over the public internet.

⚠ Different integration type
If you switch the integration type, you will overwrite the existing configuration. This action cannot be undone.

Lambda proxy integration
Send the request to your Lambda function as a structured event.

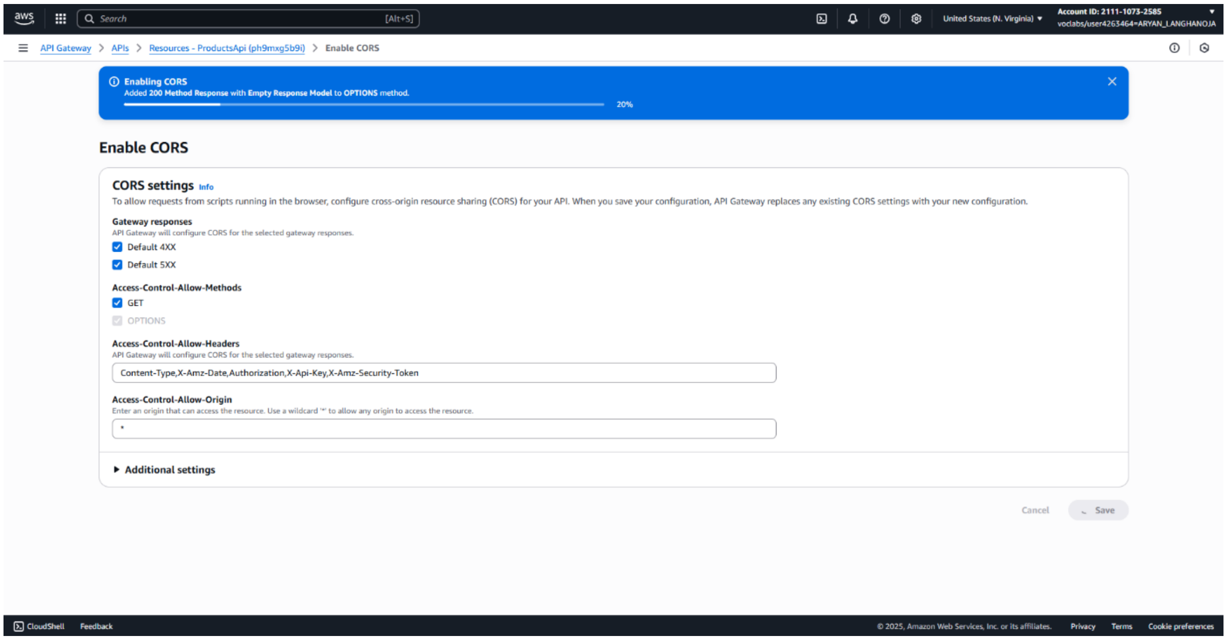
Lambda function
Provide the Lambda function name or alias. You can also provide an ARN from another account.
us-east-1 ▾ | arn:aws:lambda:us-east-1:211110732585:function:get_all_products ✕

ⓘ Grant API Gateway permission to invoke your Lambda function
When you save your changes, API Gateway updates your Lambda function's resource-based policy to allow this API to invoke it.

Execution role
arn:aws:iam::myAccount:role/myRole

Credential cache
Do not add caller credentials to cache key ▾

Integration timeout | Info



aws
Search [Alt+S]
United States (N. Virginia) ▾
Account ID: 2111-1073-2585
voclabs/user4263464=ARYAN_LANGHANOJA
API Gateway > APIs > Resources - ProductsApi (ph9mxg5b9i) > Enable CORS
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Enabling CORS
Added 200 Method Response with Empty Response Model to OPTIONS method.
20%

Enable CORS

CORS settings Info

To allow requests from scripts running in the browser, configure cross-origin resource sharing (CORS) for your API. When you save your configuration, API Gateway replaces any existing CORS settings with your new configuration.

Gateway responses
API Gateway will configure CORS for the selected gateway responses.
☑ Default 4XX
☑ Default 5XX

Access-Control-Allow-Methods
☑ GET
☑ OPTIONS

Access-Control-Allow-Headers
API Gateway will configure CORS for the selected gateway responses.
Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token

Access-Control-Allow-Origin
Enter an origin that can access the resource. Use a wildcard '*' to allow any origin to access the resource.
*

▸ Additional settings

Cancel Save

32. Configure the /on_offer integration request details.
    - Choose the **GET** method for **/products/on_offer**.
    - Choose **Integration Request**.
    - Choose **Edit**. Expand **Mapping Templates**.
    - Choose **Add mapping template**.
    - In the Content-Type box, enter the following text:

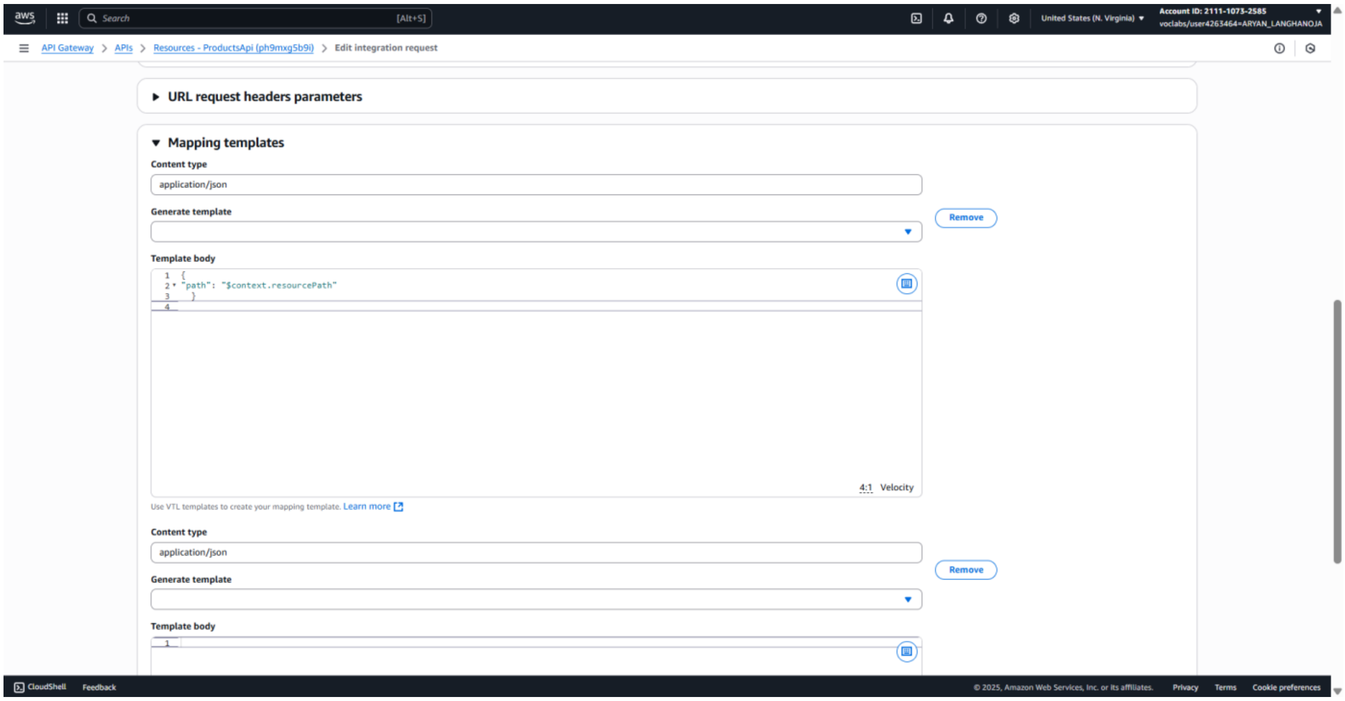
    ```
    application/json
    ```

    - Under **Generate template** choose **Method request passthrough**.
    - Replace the text with the following:

    ```
    {
    "path": "$context.resourcePath"
      }
    ```

    This will evaluate to `/products/on_offer`. For now, the code simply checks for the existence of the variable.

    - Choose **Save** at the bottom of the page.


aws
Search
[Alt+S]
United States (N. Virginia) ▾
Account ID: 2111-1073-2585
voclabs/user4263464=ARYAN_LANGHANOJA
API Gateway > APIs > Resources - ProductsApi (ph9mxg5b9i) > Edit integration request
▸ URL request headers parameters
▾ Mapping templates
Remove
Velocity
Remove
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Content type: application/json

Generate template

Template body

```
{
"path": "$context.resourcePath"
  }

```

4:1

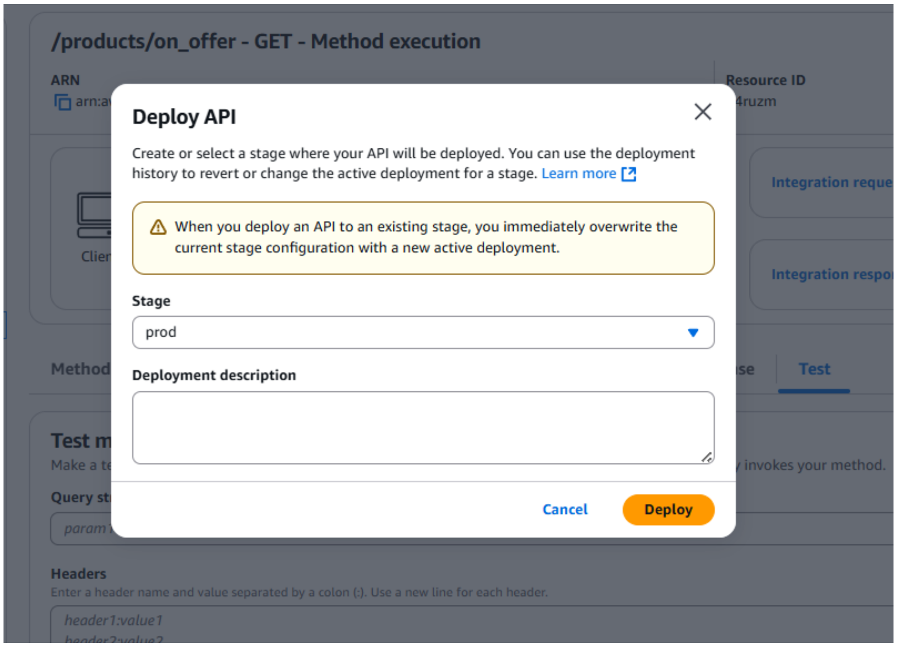
Use VTL templates to create your mapping template. Learn more

Content type: application/json

Generate template

Template body

36. Deploy the API.
    - In the **Resources** panel, choose the API root /.
    - Choose **Deploy API**.
    - For **Deployment stage**, choose **prod**, and then choose **Deploy**.



Deploy API

Create or select a stage where your API will be deployed. You can use the deployment history to revert or change the active deployment for a stage. Learn more

⚠ When you deploy an API to an existing stage, you immediately overwrite the current stage configuration with a new active deployment.

Stage: prod

Deployment description

Task 4: Creating a Lambda function for report requests in the future

37. Observe and test the Python code that you will use in the Lambda function.

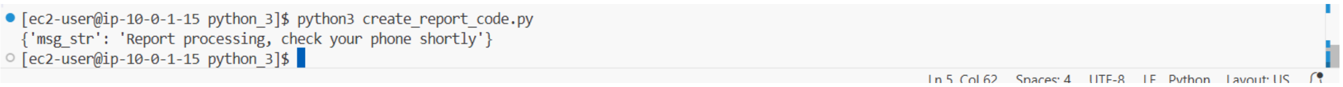
- Back in the VS Code IDE, browse to and open python_3/**create_report_code.py**.
- Notice that this code does not do much yet. It simply returns a message. In a later lab, you will implement more useful logic to actually create a report; however, this code will suffice for now.
- Run the following command in the terminal:

```
python3 create_report_code.py
```

- The output returned in the terminal looks like the following:

```
{'msg_str': 'Report processing, check your phone shortly'}
```

- Notice the capitalized "R" in the word Report. The mock data contained a lowercase "r" instead. This difference is how you can know that the website is accessing the Lambda function and not the mock data.



```
● [ec2-user@ip-10-0-1-15 python_3]$ python3 create_report_code.py
  {'msg_str': 'Report processing, check your phone shortly'}
○ [ec2-user@ip-10-0-1-15 python_3]$ 
```

38. Edit the wrapper code that you will use to create the Lambda function.

   - Browse to and open python_3/**create_report_wrapper.py**.
   - On line 5, replace the <FMI_1> placeholder with the LambdaAccessToDynamoDB **Role ARN** value.

     Tip: You may need to return to the IAM console to copy the Role ARN value.

   - Close the file by choosing **X** from the top. (Your changes are saved automatically).

39. Package the code and store it in the S3 bucket.

   - To place a copy of your code in a .zip file, run the following command:

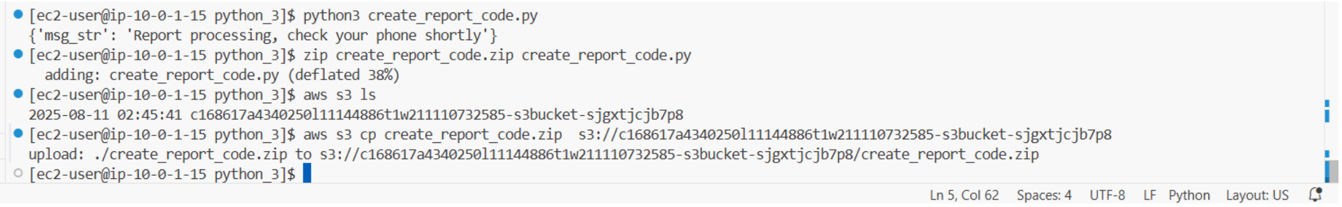
```
zip create_report_code.zip create_report_code.py
```

   - To place the .zip file in the bucket, run the following command. Replace <bucket-name> with the actual bucket name:

```
aws s3 cp create_report_code.zip  s3://<bucket-name>
```

   - Verify that the command succeeded.

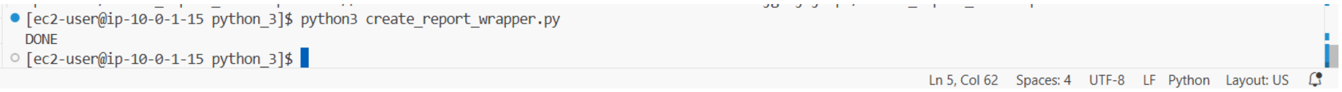
     The following is the output: `upload: ./create_report_code.zip to s3://<bucket-name>/create_report_code.zip`



```
● [ec2-user@ip-10-0-1-15 python_3]$ python3 create_report_code.py
  {'msg_str': 'Report processing, check your phone shortly'}
● [ec2-user@ip-10-0-1-15 python_3]$ zip create_report_code.zip create_report_code.py
    adding: create_report_code.py (deflated 38%)
● [ec2-user@ip-10-0-1-15 python_3]$ aws s3 ls
  2025-08-11 02:45:41 c168617a4340250l111444886t1w211110732585-s3bucket-sjgxtjcjb7p8
● [ec2-user@ip-10-0-1-15 python_3]$ aws s3 cp create_report_code.zip  s3://c168617a4340250l111444886t1w211110732585-s3bucket-sjgxtjcjb7p8
  upload: ./create_report_code.zip to s3://c168617a4340250l111444886t1w211110732585-s3bucket-sjgxtjcjb7p8/create_report_code.zip
○ [ec2-user@ip-10-0-1-15 python_3]$ 
```

41. Finally, to create the Lambda function, run the following command:
    `python3 create_report_wrapper.py`

- The output of the command is DONE, confirming that the code ran without errors.



```
● [ec2-user@ip-10-0-1-15 python_3]$ python3 create_report_wrapper.py
  DONE
○ [ec2-user@ip-10-0-1-15 python_3]$ 
```

Task 5: Configuring the REST API to invoke the Lambda function to handle reports

43. Test the existing POST method for /create_report.
    - Browse to the API Gateway console.
    - Choose the ProductsApi API, and choose the POST method for create_report.

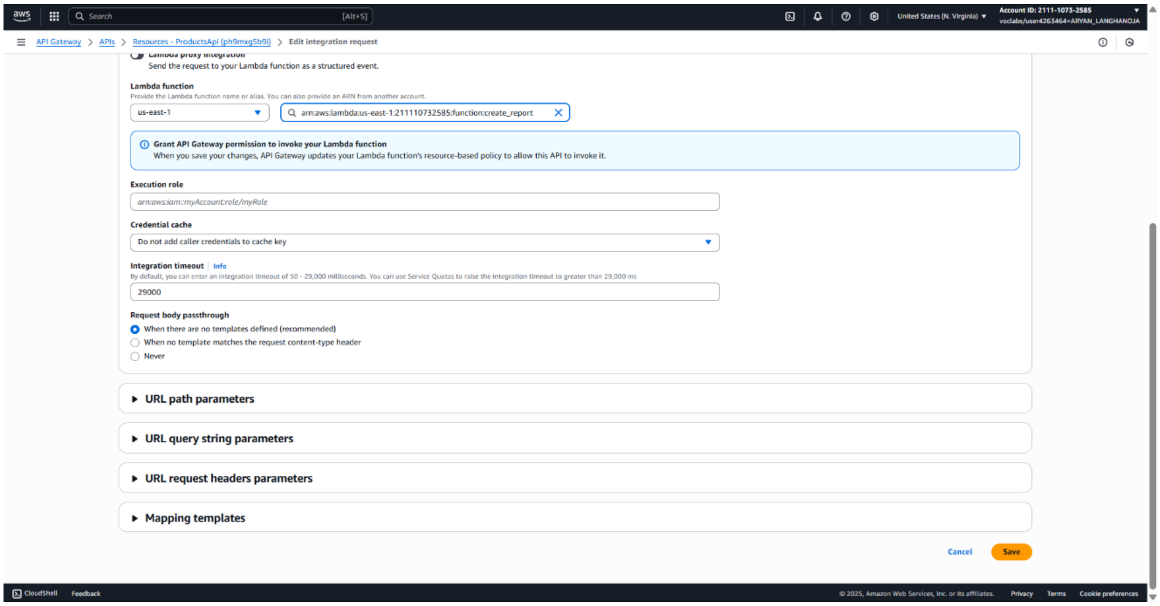
Notice on the right side of the page that the method is still accessing a "Mock Endpoint".

- Choose Test, and then choose Test at the bottom of the page.

Verify that the Response Body correctly returns the mock data (note the lowercase "r" in the results), as in the following example:

```
{
  "message_str": "report requested, check your phone shortly."
}
```

44. Replace the mock endpoint with the Lambda function.
    - Ensure that the POST method is still selected.
    - Choose Integration Request and Edit:
        - Integration type: Lambda Function
        - Lambda Region: us-east-1
        - Lambda Function: create_report
        - Choose Save.



aws
Search
[Alt+S]
United States (N. Virginia) ▾
Account ID: 2111-1073-2585
voclabs/user4263464=ARYAN_LANGHANOJA
API Gateway > APIs > Resources - ProductsApi (ph9mxg5b9i) > Edit integration request
▸ URL path parameters
▸ URL query string parameters
▸ URL request headers parameters
▸ Mapping templates
Cancel
Save
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Lambda proxy integration
Send the request to your Lambda function as a structured event.

Lambda function
Provide the Lambda function name or alias. You can also provide an ARN from another account.
us-east-1 ▾ | arn:aws:lambda:us-east-1:211110732585:function:create_report

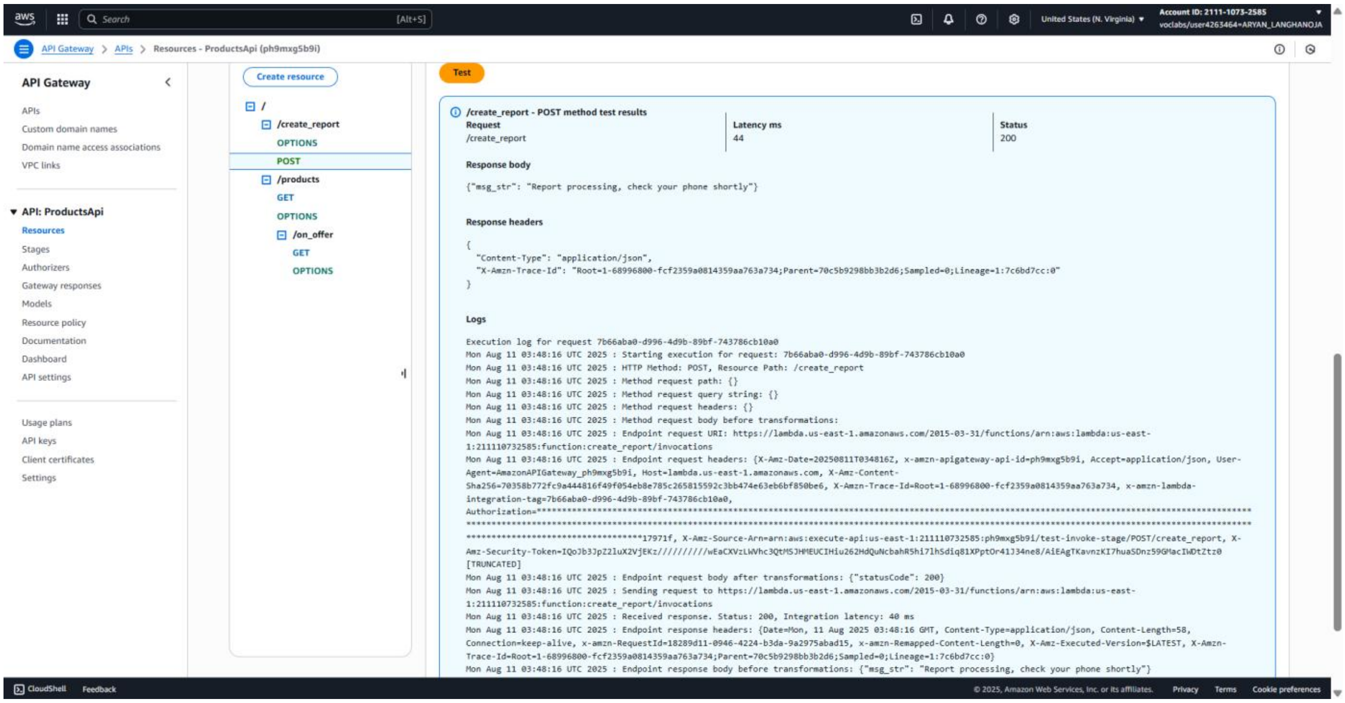
ⓘ **Grant API Gateway permission to invoke your Lambda function**
When you save your changes, API Gateway updates your Lambda function's resource-based policy to allow this API to invoke it.

Execution role
arn:aws:iam::myAccount:role/myRole

Credential cache
Do not add caller credentials to cache key ▾

Integration timeout | Info
By default, you can enter an integration timeout of 50 - 29,000 milliseconds. You can use Service Quotas to raise the integration timeout to greater than 29,000 ms
29000

Request body passthrough
● When there are no templates defined (recommended)
○ When no template matches the request content-type header
○ Never



ⓘ **/create_report - POST method test results**

Request	Latency ms	Status
/create_report	44	200

Response body

```
{"msg_str": "Report processing, check your phone shortly"}
```

Response headers

```
{
  "Content-Type": "application/json",
  "X-Amzn-Trace-Id": "Root=1-68996800-fcf2359a0814359aa763a734;Parent=70c5b9298bb3b2d6;Sampled=0;Lineage=1:7c6bd7cc:0"
}
```

Logs

```
Execution log for request 7b66aba0-d996-4d9b-89bf-743786cb10a0
Mon Aug 11 03:48:16 UTC 2025 : Starting execution for request: 7b66aba0-d996-4d9b-89bf-743786cb10a0
Mon Aug 11 03:48:16 UTC 2025 : HTTP Method: POST, Resource Path: /create_report
Mon Aug 11 03:48:16 UTC 2025 : Method request path: {}
Mon Aug 11 03:48:16 UTC 2025 : Method request query string: {}
Mon Aug 11 03:48:16 UTC 2025 : Method request headers: {}
Mon Aug 11 03:48:16 UTC 2025 : Method request body before transformations:
Mon Aug 11 03:48:16 UTC 2025 : Endpoint request URI: https://lambda.us-east-1.amazonaws.com/2015-03-31/functions/arn:aws:lambda:us-east-
1:211110732585:function:create_report/invocations
Mon Aug 11 03:48:16 UTC 2025 : Endpoint request headers: {X-Amz-Date=20250811T034816Z, x-amzn-apigateway-api-id=ph9mxg5b9i, Accept=application/json, User-
Agent=AmazonAPIGateway_ph9mxg5b9i, Host=lambda.us-east-1.amazonaws.com, X-Amz-Content-
Sha256=70358b772fc9a444816f49f054eb8e785c265815592c3bb474e63eb6bf850be6, X-Amzn-Trace-Id=Root=1-68996800-fcf2359a0814359aa763a734, x-amzn-lambda-
integration-tag=7b66aba0-d996-4d9b-89bf-743786cb10a0,
Authorization=**********************************************************************************************************************************************
************************************************************************************************************************************************************
***********************************17971f, X-Amz-Source-Arn=arn:aws:execute-api:us-east-1:211110732585:ph9mxg5b9i/test-invoke-stage/POST/create_report, X-
Amz-Security-Token=IQoJb3JpZ2luX2VjEKz//////////wEaCXVzLWVhc3QtMSJHMEUCIHiu262HdQuNcbahR5hi7lhSdiq81XPptOr41J34ne8/AiEAgTKavnzKI7huaSDnz59GMacIWDtZtz0
[TRUNCATED]
Mon Aug 11 03:48:16 UTC 2025 : Endpoint request body after transformations: {"statusCode": 200}
Mon Aug 11 03:48:16 UTC 2025 : Sending request to https://lambda.us-east-1.amazonaws.com/2015-03-31/functions/arn:aws:lambda:us-east-
1:211110732585:function:create_report/invocations
Mon Aug 11 03:48:16 UTC 2025 : Received response. Status: 200, Integration latency: 40 ms
Mon Aug 11 03:48:16 UTC 2025 : Endpoint response headers: {Date=Mon, 11 Aug 2025 03:48:16 GMT, Content-Type=application/json, Content-Length=58,
Connection=keep-alive, x-amzn-RequestId=18289d11-0946-4224-b3da-9a2975abad15, x-amzn-Remapped-Content-Length=0, X-Amz-Executed-Version=$LATEST, X-Amzn-
Trace-Id=Root=1-68996800-fcf2359a0814359aa763a734;Parent=70c5b9298bb3b2d6;Sampled=0;Lineage=1:7c6bd7cc:0}
Mon Aug 11 03:48:16 UTC 2025 : Endpoint response body before transformations: {"msg_str": "Report processing, check your phone shortly"}
```

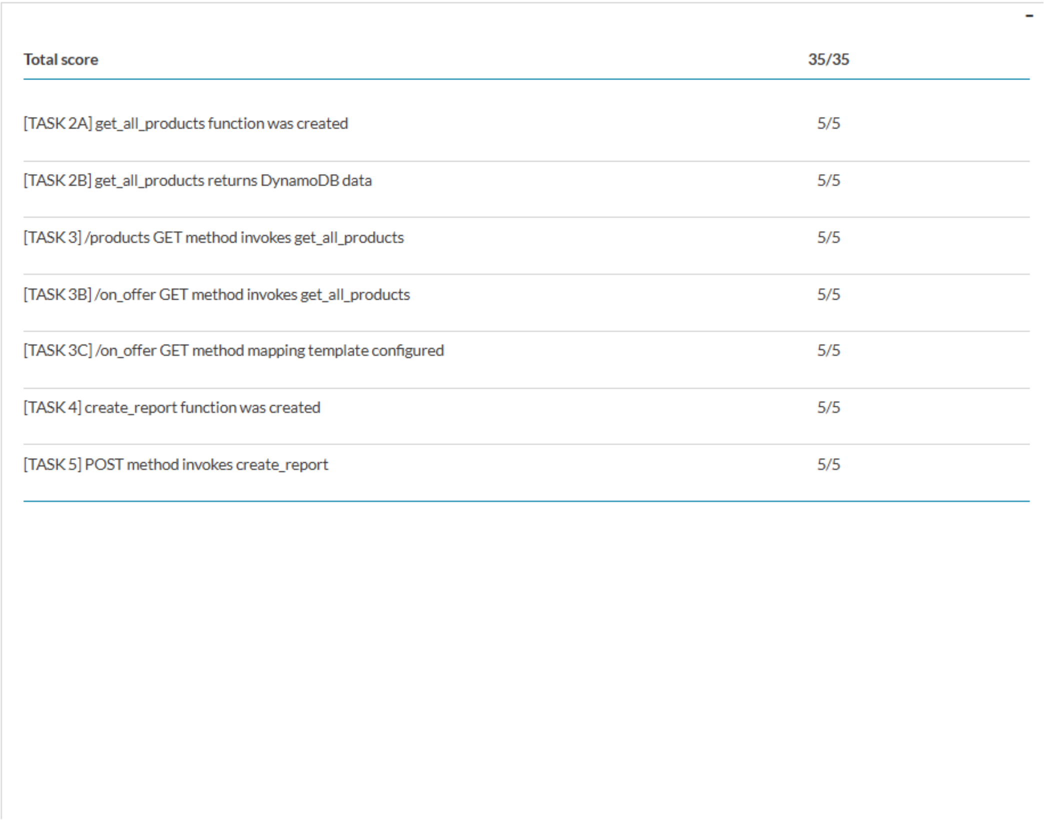
45. Deploy the API.
    - In the Resources panel, choose the API root /.
    - Choose Deploy API.
    - For Deployment stage, choose prod, and then choose Deploy.

Congratulations! You have successfully updated the REST API so that it invokes the create_report Lambda function.

Conclusion:-

- In this Lab I Learned the AWS Lambda Function Service.
- I had integrated the REST API of GET Method /product with lambda function using python.
- I had intgrrated the REST API of GET Method /product/on_offer with lambda function using python.
- I had integrated the REST API of POST Method /create_report with lambda functon using python.
- I had re-deployed that API

Result :-



Total score	35/35
[TASK 2A] get_all_products function was created	5/5
[TASK 2B] get_all_products returns DynamoDB data	5/5
[TASK 3] /products GET method invokes get_all_products	5/5
[TASK 3B] /on_offer GET method invokes get_all_products	5/5
[TASK 3C] /on_offer GET method mapping template configured	5/5
[TASK 4] create_report function was created	5/5
[TASK 5] POST method invokes create_report	5/5