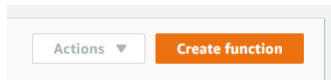


While there are a couple other options, it's almost impossible to talk about "Serverless" within AWS without "Lambda" coming into the conversation.

AWS Lambda provides an easy way to execute code without maintaining the underlying hardware. Where VMs abstract away much of the hardware, and Containers abstract away much of the operating system, Serverless Functions abstract away much of the container setup and orchestration. In a Lambda application you will specify the amount of memory, the permissions, the runtime (Python, Java, C#, etc), how the function should be invoked, and the Lambda orchestration takes care of making sure it all happens.

In this exercise we'll create a basic lambda function in Python, and just run it from within the Lambda UI.

Log into the AWS Console, open the "Lambda" service, and hit "Create Function"

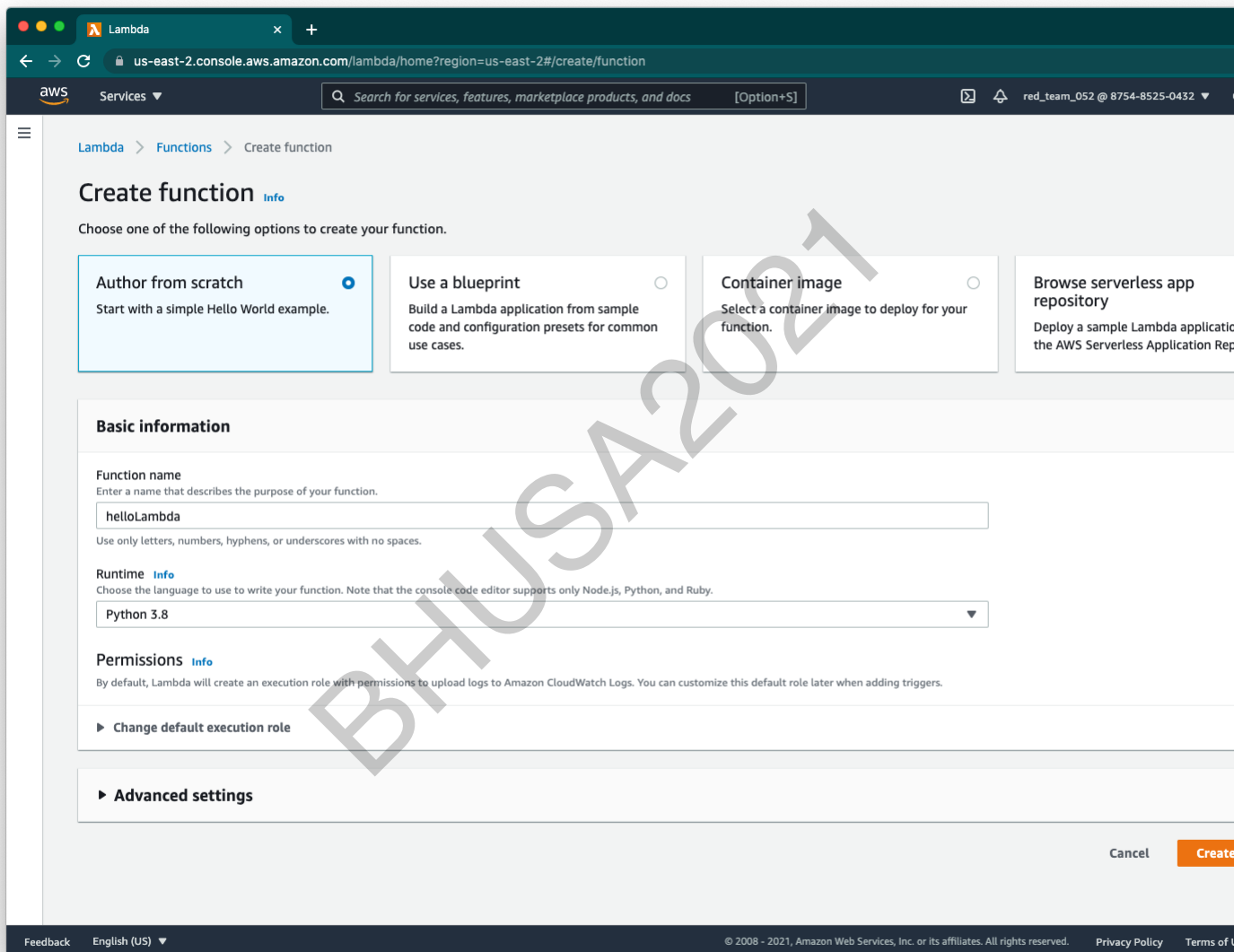


Leave the "Author from scratch" radio button checked

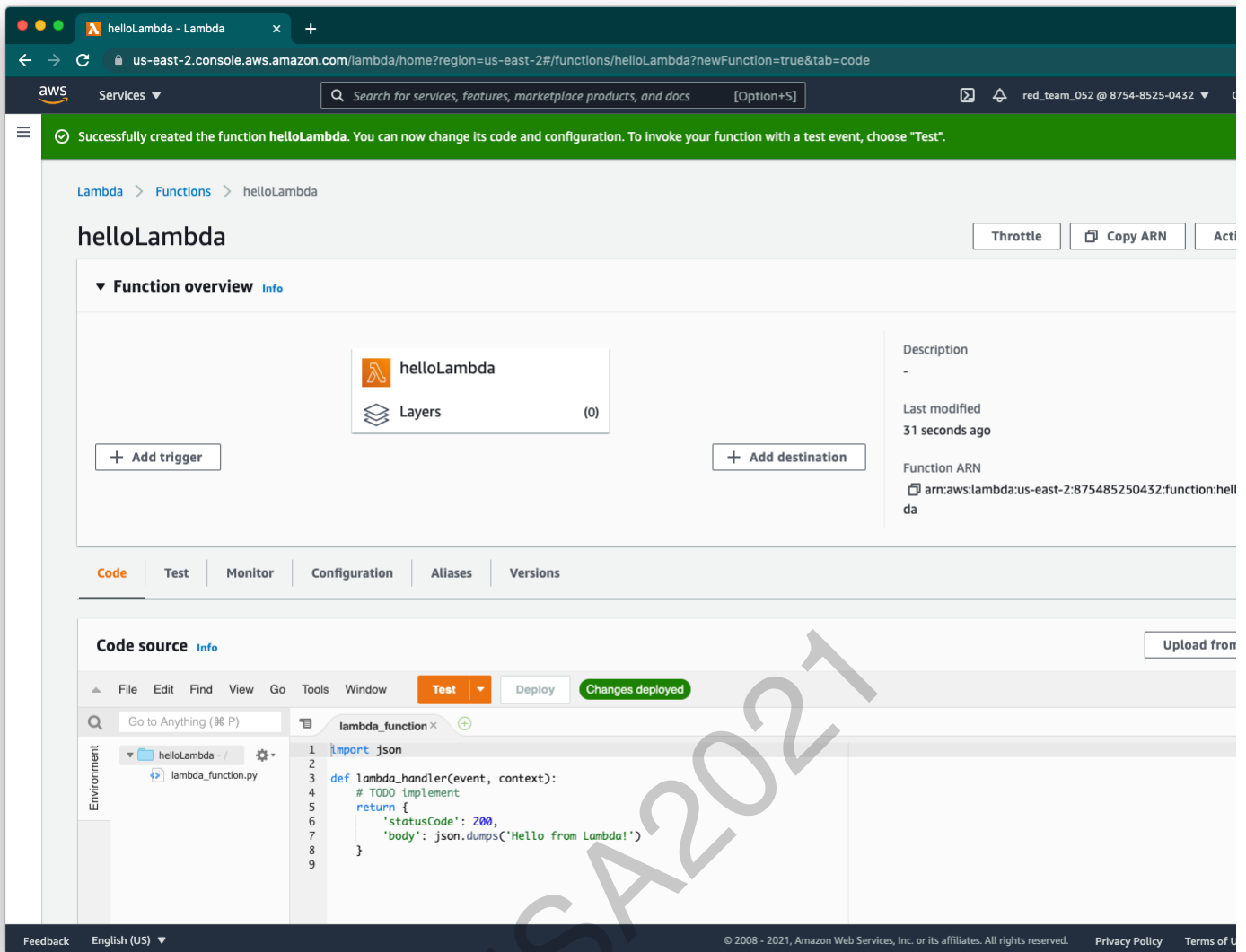
Give it the function name of "helloLambda"

Switch the runtime to "Python3.8"

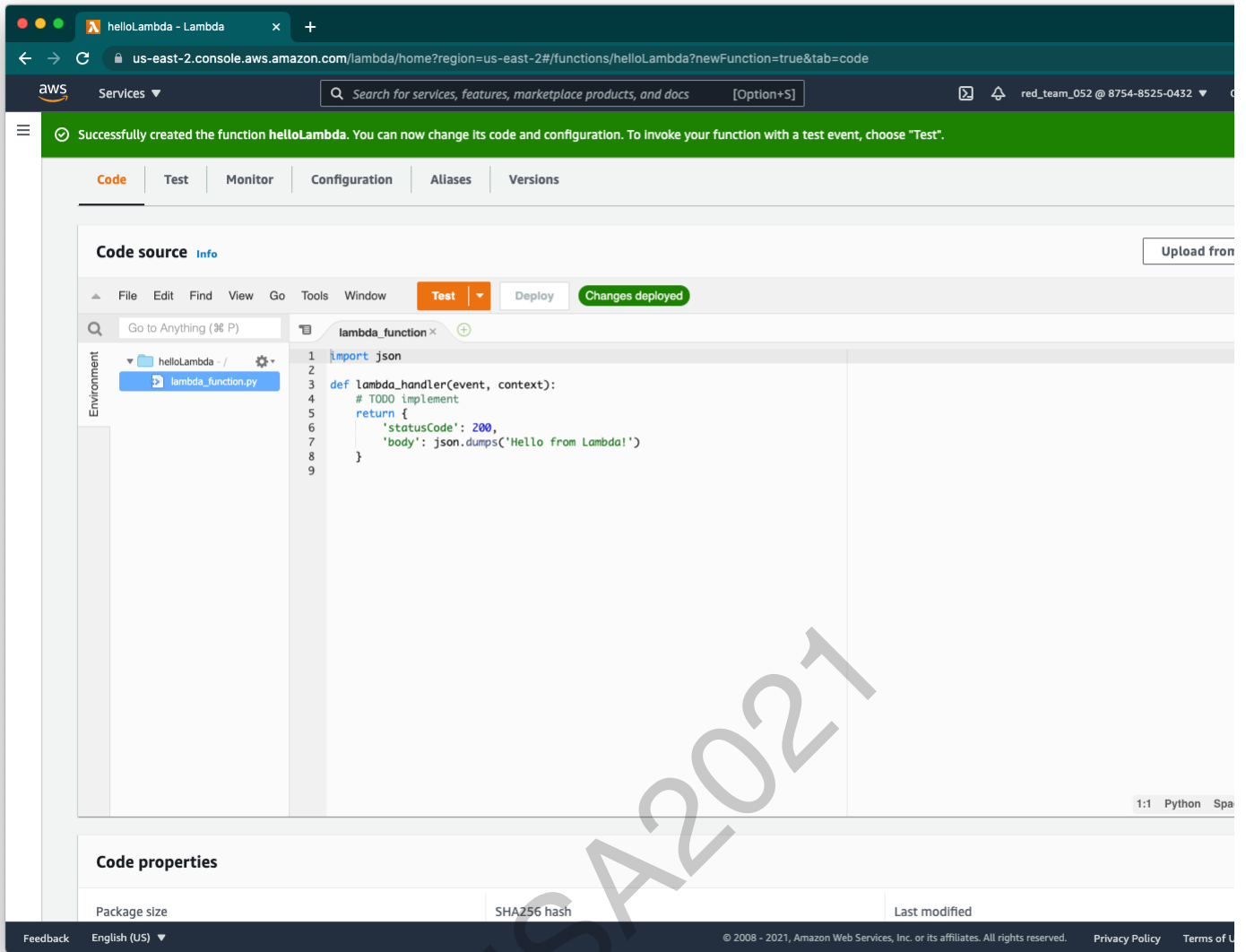
Click the "Create Function" button



You will be dropped into a function configuration interface similar to the following...

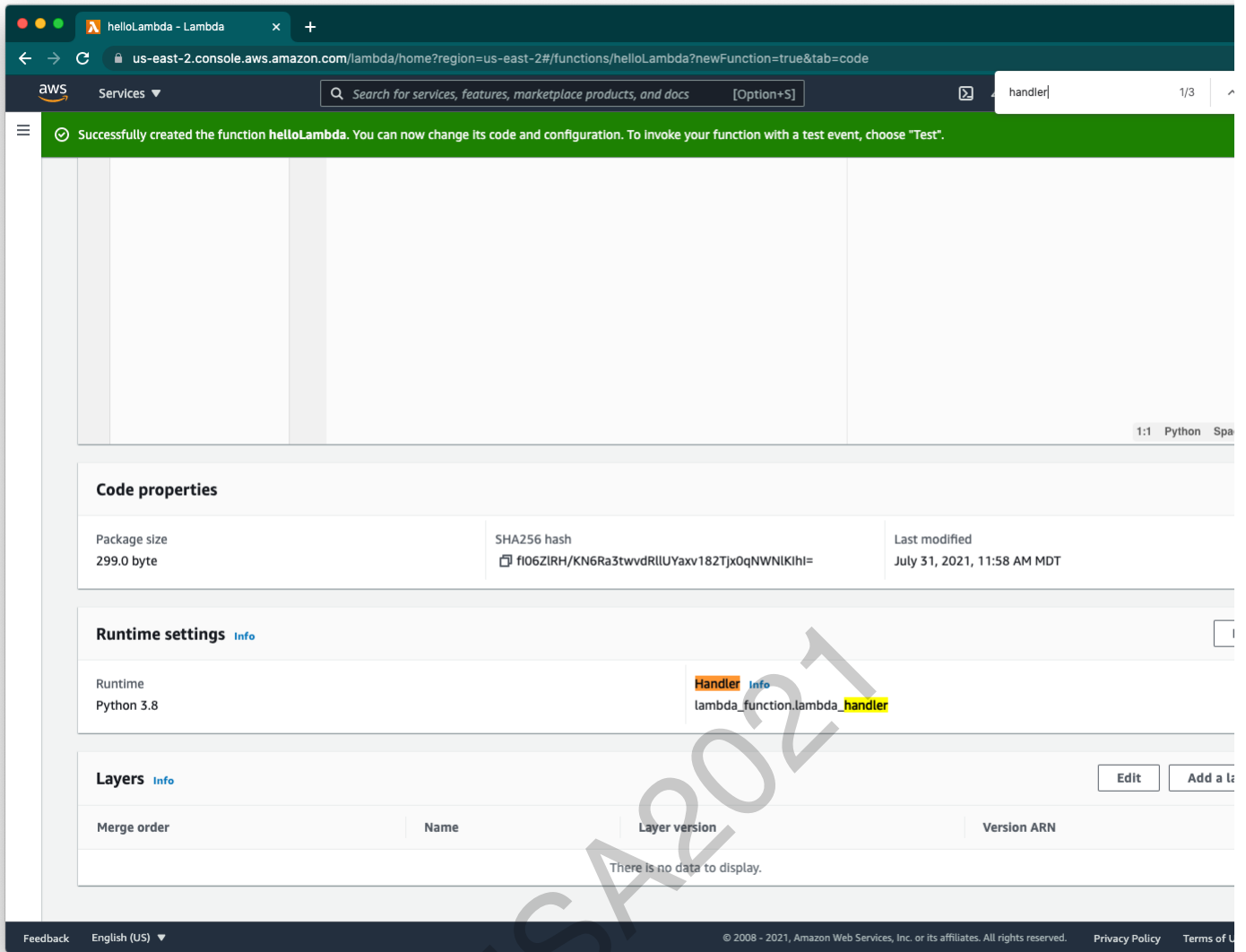


Double click on the "lambda_function.py" file to see a limited code editor with the skeleton of a lambda function...



Scroll down a bit more...

A setting to note is the "Handler", which instructs Lambda which python function within your script to execute on invocation...



Scroll back up to the top of the page...

Click on the "Configuration" link...

Under the "General configuration" section, note the default settings of 128MB memory and a 3-second timeout.

aws Services ▾ Search for services, features, marketplace products, and docs [Alt+S] red_team_020 @ 1188

Lambda > Functions > helloLambda

helloLambda

Throttle Co

▼ Function overview Info

helloLambda

Layers (0)

+ Add trigger

Description

-

Last modified

6 minutes ago

Function ARN

arn:aws:lambda:us-east-2:118667365703:fu

+ Add destination

Code | Test | Monitor | **Configuration** | Aliases | Versions

General configuration

Triggers

Permissions

Destinations

Environment variables

Tags

General configuration Info

Description	Memory (MB)	Timeout
-	128	0 min 3 sec

AWS Compute Optimizer

Opt in to see memory recommendations for your Lambda functions. [View details](#)

If you expect your function to take longer than 3 seconds to execute you can increase this.

Set the Timeout to 10 seconds by click the "Edit" button and then changing the "sec" field to 10...

aws Services ▾ Search for services, features, marketplace products, and docs

Lambda > Functions > helloLambda > Edit basic settings

Edit basic settings

Basic settings Info

Description - optional

Memory (MB) Info

Your function is allocated CPU proportional to the memory configured.

128 MB

Set memory to between 128 MB and 10240 MB

Timeout

0 min 10 sec

Execution role

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

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.

service-role/helloLambda-role-anzf6kfm

[View the helloLambda-role-anzf6kfm role](#) on the IAM console.

Cancel Save

Now click the "Save" button and verify the new timeout setting...

aws Services Search for services, features, marketplace products, and docs [Alt+S] red_team_020 @ 1186

Your changes have been saved.

Lambda > Functions > helloLambda

helloLambda

Throttle Co

▼ Function overview Info

helloLambda

Layers (0)

+ Add trigger

+ Add destination

Description -

Last modified 12 seconds ago

Function ARN [arn:aws:lambda:us-east-2:118667365703:fu](#)

Code Test Monitor Configuration Aliases Versions

General configuration Info

Description -	Memory (MB) 128	Timeout 0 min 10 sec
---------------	-----------------	----------------------

AWS Compute Optimizer
Opt in to see memory recommendations for your Lambda functions. [View details](#)

Triggers

Permissions

Destinations

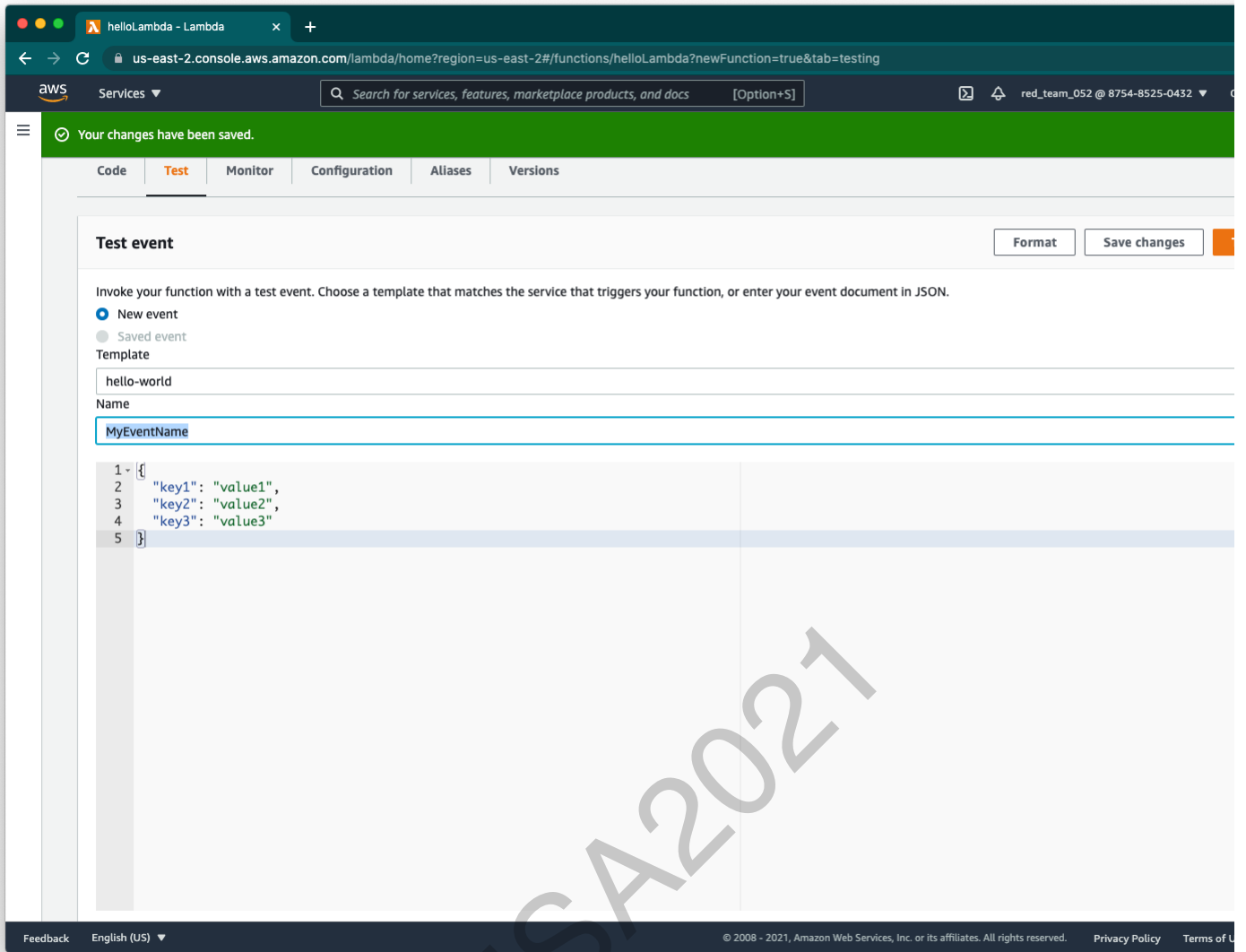
Environment variables

Tags

Next, create a test case by clicking the "Test" link...

Set the "Name" field to "MyEventName"...

BHUSA2021



Click the "Test" button and then we should see output similar to this...

The screenshot shows the AWS Lambda console for a function named 'helloLambda'. The 'Test' tab is active, displaying a green notification: 'Execution result: succeeded (logs)' with a 'Details' link. Below this, the 'Test event' section is visible, featuring a 'Format' button and a 'Save changes' button. The 'Test event' section includes a radio button for 'New event' (selected) and a 'Saved event' option. A 'Template' dropdown is set to 'hello-world'. The 'Name' field contains 'MyEventName'. A code editor shows a JSON event document:

```
1 {
2   "key1": "value1",
3   "key2": "value2",
4   "key3": "value3"
5 }
```

The footer of the console shows 'Feedback', 'English (US)', and copyright information: '© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of U'.

Click the "Details" link and we should see output similar to the following...

The screenshot shows the AWS Lambda console interface. At the top, there's a navigation bar with the AWS logo, a search bar, and the user's account information. Below that, a green banner indicates "Your changes have been saved." The main content area is divided into tabs: Code, Test, Monitor, Configuration, Aliases, and Versions. The "Test" tab is active, showing the execution result for a function named "helloLambda".

Execution result: succeeded (logs)

Details

The area below shows the result returned by your function execution. [Learn more](#) about returning results from your function.

```
{
  "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}
```

Summary

Code SHA-256	Request ID
f106ZIRH/KN6Ra3twvdRIUyaxv182Tjx0qNWNKIhI=	2fcb8a26-3897-44ac-9a74-7ccb447a73f2
Init duration	Duration
117.33 ms	1.23 ms
Billed duration	Resources configured
2 ms	128 MB
Max memory used	
51 MB	

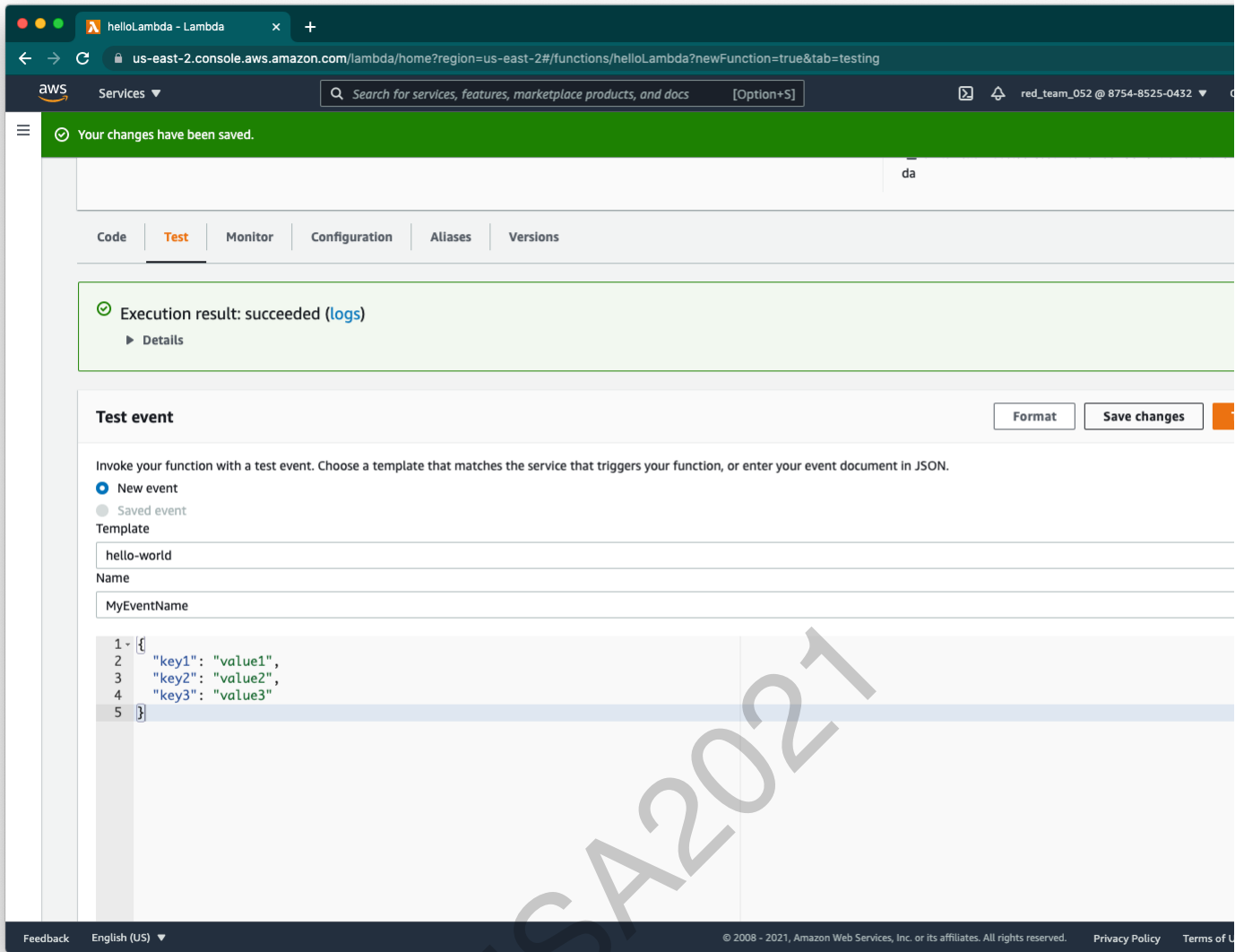
Log output

The section below shows the logging calls in your code. [Click here](#) to view the corresponding CloudWatch log group.

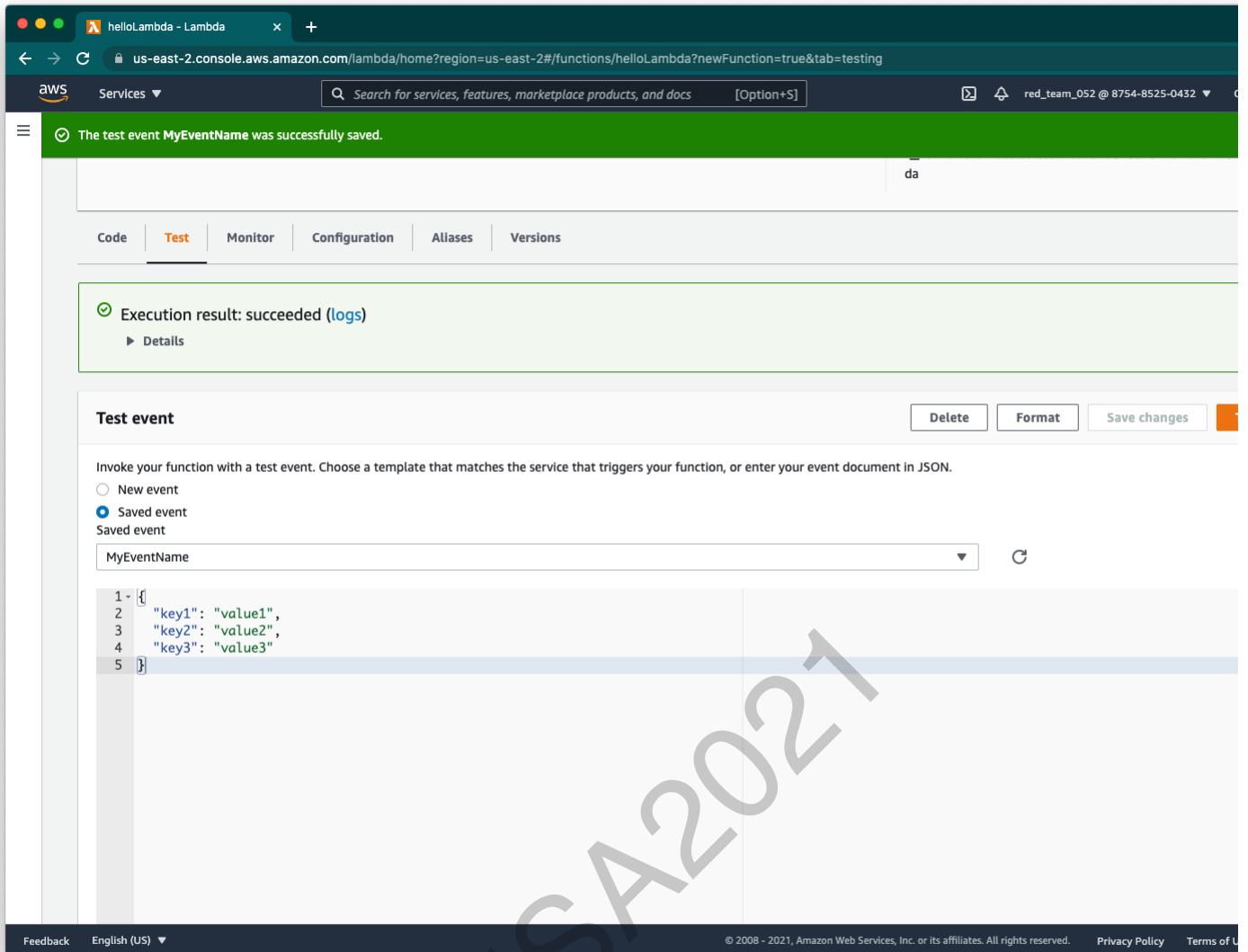
```
START RequestId: 2fcb8a26-3897-44ac-9a74-7ccb447a73f2 Version: $LATEST
END RequestId: 2fcb8a26-3897-44ac-9a74-7ccb447a73f2
REPORT RequestId: 2fcb8a26-3897-44ac-9a74-7ccb447a73f2  Duration: 1.23 ms    Billed Duration: 2 ms    Memory Size: 128 MB    Max Memory Used: 51 MB    Init Duration: 117.33 ms
```

At the bottom of the console, there's a "Test event" section with buttons for "Format" and "Save changes".

If the test case worked as expected, then click the "Save changes" button...



The screen should now change slightly to reflect the saved test...



Click the "Code" link...

Double Click the "lambda_function.py" file...

Paste the following script into the "Function code" editor for the "lambda_function.py" file, which calculates prime numbers:

```
import math

def genPrimes(context):
    primes = []
    count = 3
    while True:
        isprime = True
        for x in range(2, int(math.sqrt(count) + 1)):
            if context.get_remaining_time_in_millis() < 1000:
                return primes
            if count % x == 0:
                isprime = False
                break
        if isprime:
            primes.append(count)
            count += 1

def lambda_handler(event, context):
    primes = genPrimes(context)
    return primes
```

Click the "Deploy" button to save the changes...

```
1 import math
2
3 def genPrimes(context):
4     primes = []
5     count = 3
6     while True:
7         isprime = True
8         for x in range(2, int(math.sqrt(count) + 1)):
9             if context.get_remaining_time_in_millis() < 1000:
10                return primes
11            if count % x == 0:
12                isprime = False
13                break
14            if isprime:
15                primes.append(count)
16                count += 1
17
18 def lambda_handler(event, context):
19     primes = genPrimes(context)
20     return primes
```

Click the "Test" button...

The test event MyEventName was successfully saved.

Lambda > Functions > helloLambda

helloLambda

Throttle Copy

Function overview Info

helloLambda (0) Layers

+ Add trigger + Add destination

Description -

Last modified 7 minutes ago

Function ARN arn:aws:lambda:us-east-2:118667365703:fun

Code Test Monitor Configuration Aliases Versions

Code source Info

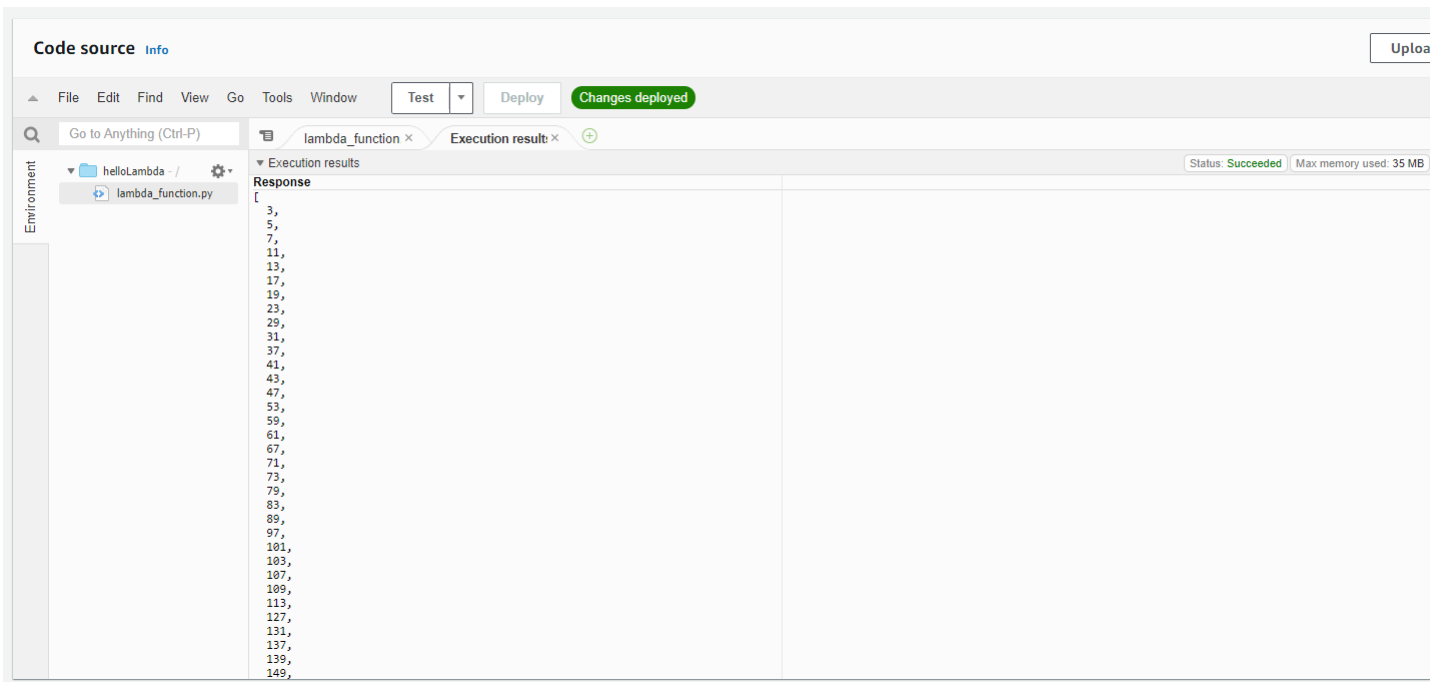
File Edit Find View Go Tools Window Test Deploy Changes not deployed

Go to Anything (Ctrl-P)

lambda_function Run the selected Lambda test (Ctrl-Shift-I)

```
1 import math
2
3 def genPrimes(context):
4     primes = []
5     count = 3
6     while True:
7         isprime = True
8         for x in range(2, int(math.sqrt(count) + 1)):
9             if context.get_remaining_time_in_millis() < 1000:
10                return primes
11            if count % x == 0:
12                isprime = False
13                break
14            if isprime:
15                primes.append(count)
16                count += 1
17
18 def lambda_handler(event, context):
19     primes = genPrimes(context)
20     return primes
```

After we reach the max execution time of 10 seconds, we should see the following output...



Memory and compute cores are linked, by increasing memory the function will be given additional compute.

You may try increasing the memory to see how many more primes the function is able to execute in the same amount of time.

This is a very simple example of how Lambda works. The power of Lambda comes in the scalability and number of ways functions can be invoked.

References

Check out the following references for more information:

- Run a Serverless "Hello, World!" - <https://aws.amazon.com/getting-started/tutorials/run-serverless-code/>
- Create a Simple Lambda Function - <https://docs.aws.amazon.com/lambda/latest/dg/get-started-create-function.html>
- Getting Started - AWS Lambda - <https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html>