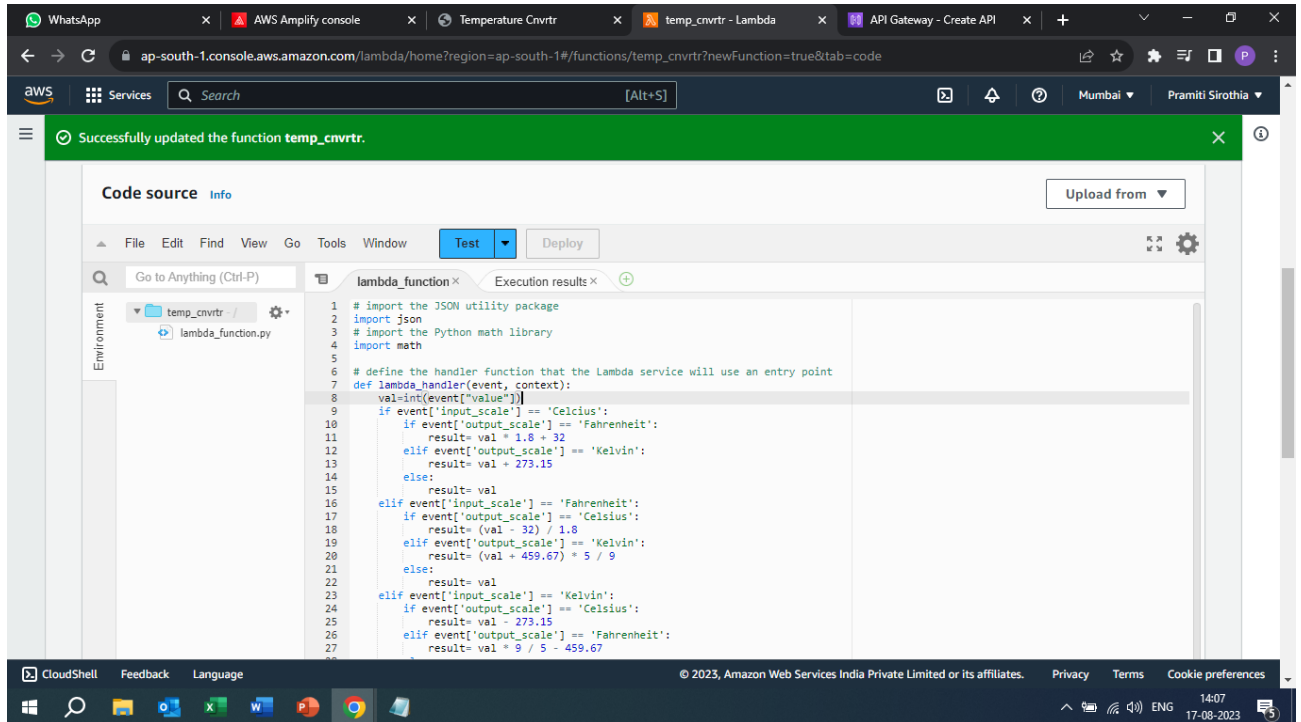
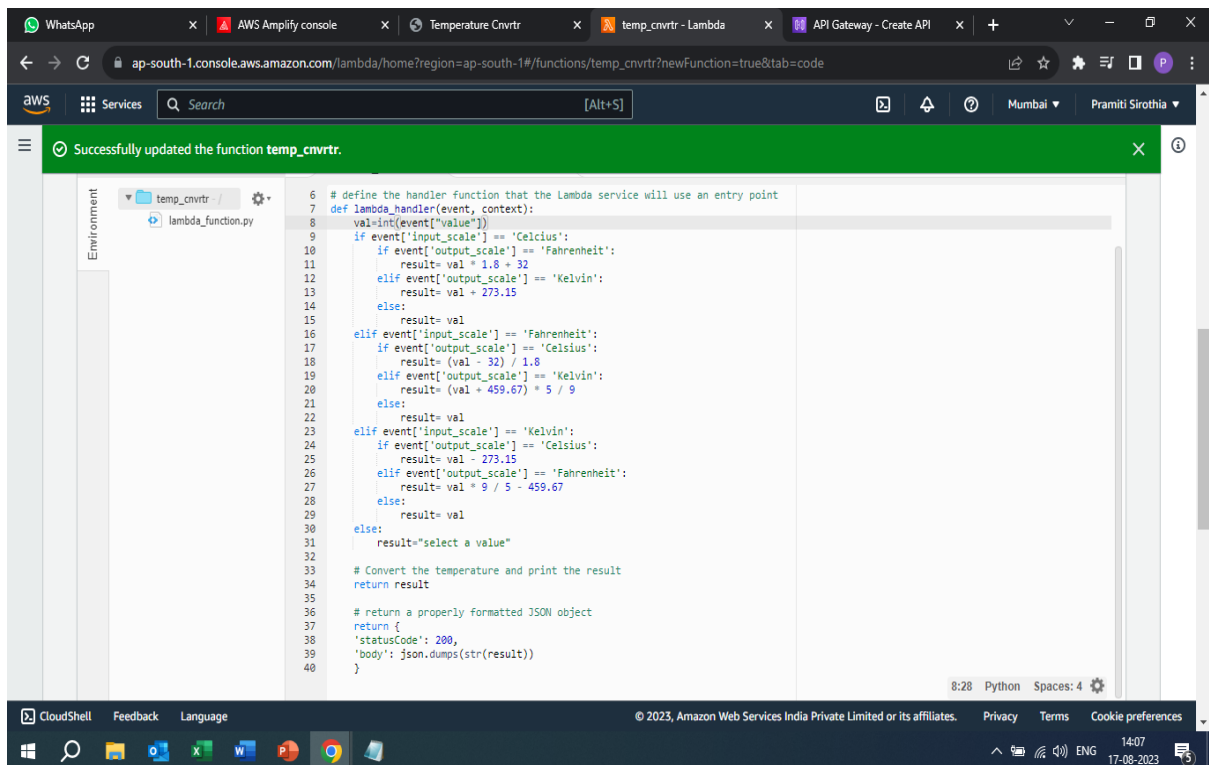


Lambda function



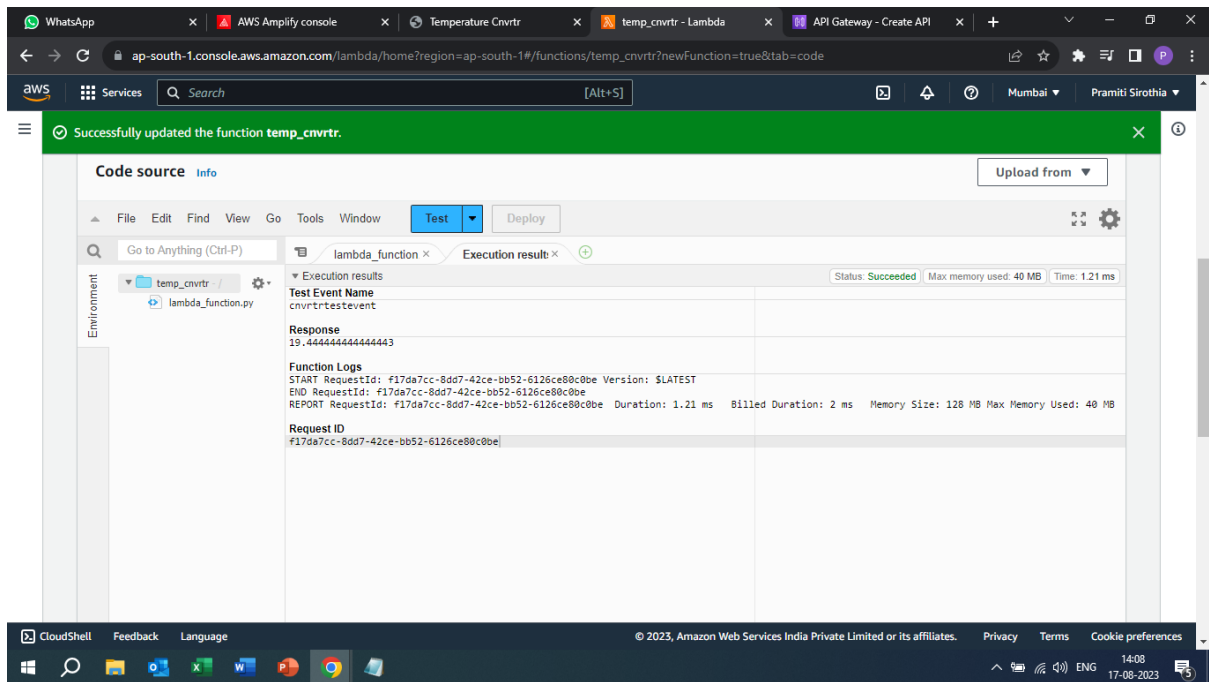
The screenshot shows the AWS Lambda console interface for the function `temp_cnvrtr`. A green notification bar at the top states "Successfully updated the function temp_cnvrtr." The "Code source" tab is active, displaying the Python code for the `lambda_handler` function. The code imports `json` and `math` libraries and defines a handler function that takes an event and context as input. It processes the `input_scale` and `output_scale` to convert temperatures between Celsius, Fahrenheit, and Kelvin. The console includes a file explorer on the left showing the `temp_cnvrtr` folder and `lambda_function.py` file. The bottom status bar indicates the function is using Python 3.11 and has 4 spaces configured.

```
1 # import the JSON utility package
2 import json
3 # import the Python math library
4 import math
5
6 # define the handler function that the Lambda service will use as an entry point
7 def lambda_handler(event, context):
8     val=int(event["value"])
9     if event['input_scale'] == 'Celcius':
10         if event['output_scale'] == 'Fahrenheit':
11             result= val * 1.8 + 32
12         elif event['output_scale'] == 'Kelvin':
13             result= val + 273.15
14         else:
15             result= val
16     elif event['input_scale'] == 'Fahrenheit':
17         if event['output_scale'] == 'Celsius':
18             result= (val - 32) / 1.8
19         elif event['output_scale'] == 'Kelvin':
20             result= (val + 459.67) * 5 / 9
21         else:
22             result= val
23     elif event['input_scale'] == 'Kelvin':
24         if event['output_scale'] == 'Celsius':
25             result= val - 273.15
26         elif event['output_scale'] == 'Fahrenheit':
27             result= val * 9 / 5 - 459.67
28         else:
29             result= val
```

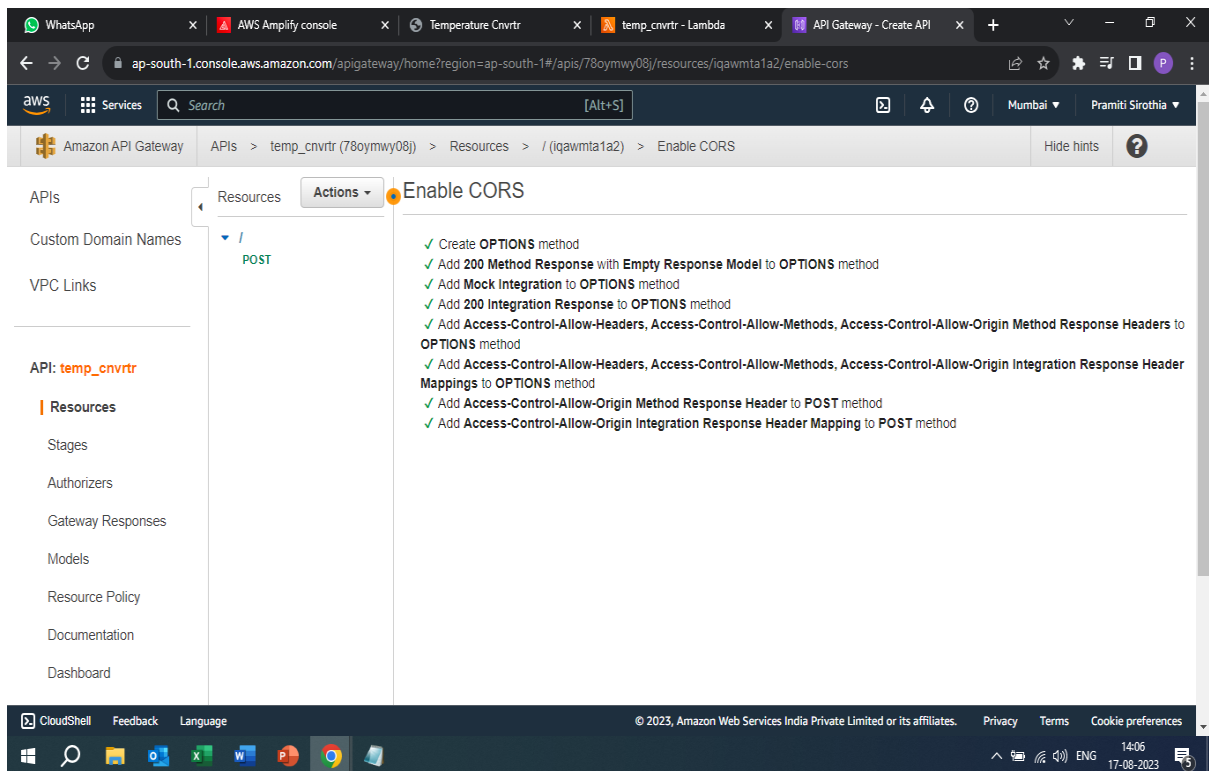


This screenshot shows the same AWS Lambda console interface for the `temp_cnvrtr` function, but with the code updated to include a return statement. The `lambda_handler` function now returns a JSON object with a `statusCode` of 200 and the converted temperature in the `body`. The code is shown from line 6 to 40. The environment section on the left remains the same, showing the `temp_cnvrtr` folder and `lambda_function.py` file. The bottom status bar shows the function is using Python 3.11 and has 4 spaces configured.

```
6 # define the handler function that the Lambda service will use as an entry point
7 def lambda_handler(event, context):
8     val=int(event["value"])
9     if event['input_scale'] == 'Celcius':
10         if event['output_scale'] == 'Fahrenheit':
11             result= val * 1.8 + 32
12         elif event['output_scale'] == 'Kelvin':
13             result= val + 273.15
14         else:
15             result= val
16     elif event['input_scale'] == 'Fahrenheit':
17         if event['output_scale'] == 'Celsius':
18             result= (val - 32) / 1.8
19         elif event['output_scale'] == 'Kelvin':
20             result= (val + 459.67) * 5 / 9
21         else:
22             result= val
23     elif event['input_scale'] == 'Kelvin':
24         if event['output_scale'] == 'Celsius':
25             result= val - 273.15
26         elif event['output_scale'] == 'Fahrenheit':
27             result= val * 9 / 5 - 459.67
28         else:
29             result= val
30     else:
31         result="select a value"
32
33     # Convert the temperature and print the result
34     return result
35
36 # return a properly formatted JSON object
37 return {
38     'statusCode': 200,
39     'body': json.dumps(str(result))
40 }
```



API CORS Configuration



IAM Policies

The screenshot displays the AWS IAM console interface for creating a policy. The browser address bar shows the URL: `us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/roles/details/temp_cnvrtr-role-4togc0jv/createPolicy?step=addPermissions`. The console shows the 'Specify permissions' step, with a sidebar indicating 'Step 1: Specify permissions' and 'Step 2: Review and create'.

The main content area is titled 'Specify permissions' and includes a sub-header 'Policy editor'. The policy document is shown in the 'Visual' tab, with the following JSON structure:

```
1 - {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "VisualEditor0",
6       "Effect": "Allow",
7       "Action": [
8         "dynamodb:PutItem",
9         "dynamodb:DeleteItem",
10        "dynamodb:GetItem",
11        "dynamodb:Scan",
12        "dynamodb:Query",
13        "dynamodb:UpdateItem"
14      ],
15      "Resource": "arn:aws:dynamodb:ap-south-1:315618505227:table/temp_cnvrtr"
16    }
17  ]
18 }
19 }
```

The 'Add actions' section on the right shows a search bar and a list of available services, including AMP, API Gateway, API Gateway V2, ASC, Access Analyzer, Account, Activate, Alexa for Business, Amplify, Amplify Admin, and Amplify UI Builder.

API Creation

The image displays two screenshots of the AWS API Gateway console interface.

Top Screenshot: Method Execution Page

- URL:** `ap-south-1.console.aws.amazon.com/apigateway/home?region=ap-south-1#/apis/78oywmwy08j/resources/iqawmta1a2/methods/POST`
- Page Title:** Method Execution / - POST - Method Test
- Left Sidebar:** Shows the API `temp_cnvrtr` and its resources. The `POST` method is selected under the `/` resource.
- Main Content Area:** Displays details for the `POST` method.
 - Path:** `/`. Note: No path parameters exist for this resource.
 - Query Strings:** Note: No query string parameters exist for this method.
 - Headers:** Note: No header parameters exist for this method.
 - Stage Variables:** Note: No stage variables exist for this method.
 - Request Body:** (Empty field)
 - Request:** `/`
 - Status:** 200
 - Latency:** 272 ms
 - Response Body:** `19.444444444444443`
 - Response Headers:**

```
{ "Access-Control-Allow-Origin": ["*"], "Content-Type": ["application/json"], "X-Amzn-Trace-Id": ["Root=1-64ddd9b-1a2c8180b3a9c11e88480103;Sampled=0;Lineage=d60d2291:0"] }
```
 - Logs:** Execution log for request `59c2ef77-e9b3-46dc-b1ce-4c5544086e5f`. Log entry: `Thu Aug 17 08:43:07 UTC 2023 : Starting execution for request: 59c2ef77-e9b3-46dc-b1ce-4c5544086e5f Thu Aug 17 08:43:07 UTC 2023 : HTTP Method: POST, Res`

Bottom Screenshot: dev Stage Editor Page

- URL:** `ap-south-1.console.aws.amazon.com/apigateway/home?region=ap-south-1#/apis/78oywmwy08j/stages/dev`
- Page Title:** dev Stage Editor
- Left Sidebar:** Shows the API `temp_cnvrtr` and its stages. The `dev` stage is selected.
- Main Content Area:** Displays settings for the `dev` stage.
 - Invoke URL:** `https://78oywmwy08j.execute-api.ap-south-1.amazonaws.com/dev`
 - Settings Tab:** Active. Sub-tabs include `Canary`, `Cache Settings`, `Default Method Throttling`, `Stage Variables`, `SDK Generation`, `Export`, `Deployment History`, and `Documentation History`.
 - Cache Settings:** `Enable API cache` is unchecked.
 - Default Method Throttling:** `Enable throttling` is checked. Settings: `Rate` is `10000` requests per second, and `Burst` is `5000` requests.

Application Deployment

The image consists of two screenshots. The top screenshot shows the AWS Amplify console interface. The browser address bar displays `ap-south-1.console.aws.amazon.com/amplify/home?region=ap-south-1#/d211h3tgg158q3`. The console shows a deployment status of "Deployment successfully completed." with a green progress bar at 100%. Below this, the domain is listed as `https://dev.d211h3tgg158q3.amplifyapp.com` and the last deployment is dated 8/17/2023, 3:00:36 PM. A drag-and-drop area is present with a "Choose files" button. The bottom screenshot shows the deployed application in a web browser. The URL is `dev.d211h3tgg158q3.amplifyapp.com`. The application is titled "Temperature Converter" and features a form with two input fields: "From" (containing "98") and "To" (containing "36.666666666666664"). The "From" field is set to "Fahrenheit" and the "To" field is set to "Celsius". A green "Convert" button is located below the input fields.

dev

Deployment successfully completed.

100%

Domain
<https://dev.d211h3tgg158q3.amplifyapp.com>

Last deployment
8/17/2023, 3:00:36 PM

Drag and drop your project's build output directory or zip file here to update your app, or, [choose another method](#).

Choose files

CloudShell Feedback Language © 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences

temp_cnvrtr - Lambda Temperature Cnvrtr

dev.d211h3tgg158q3.amplifyapp.com

Temperature Converter

From	To
98	Fahrenheit
36.666666666666664	Celsius
Convert	