

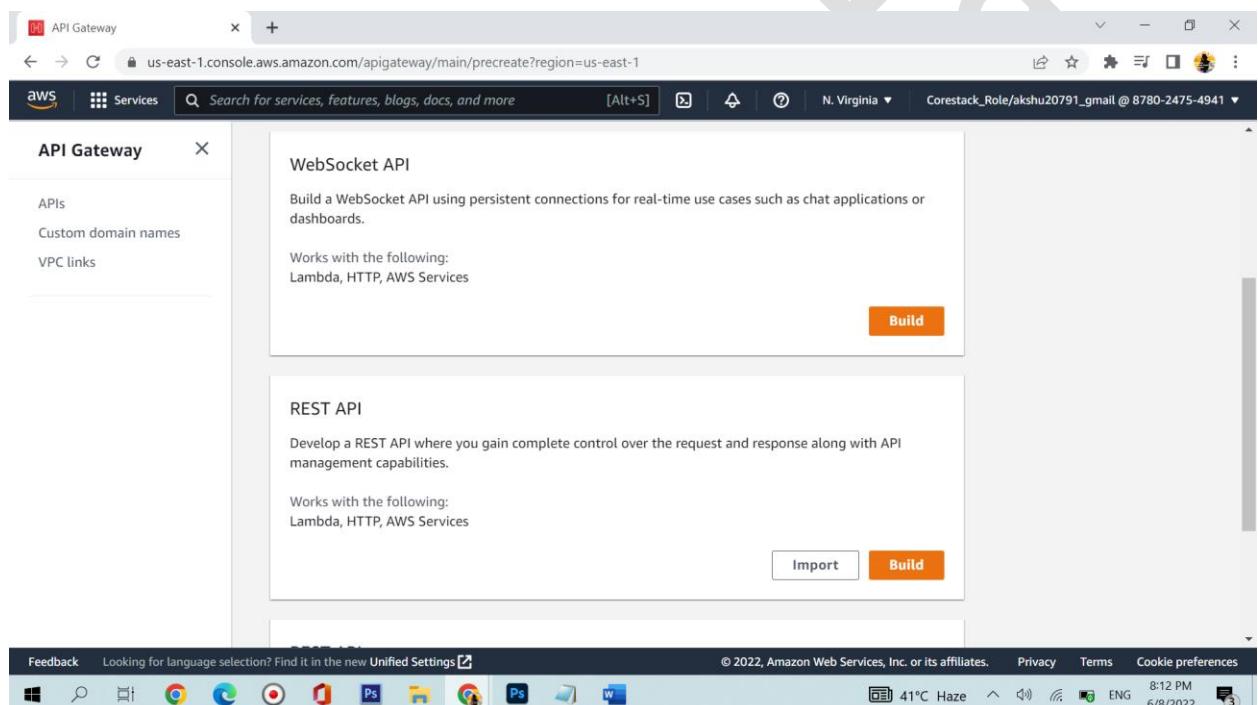
API Gateway

Amazon API Gateway is a fully managed service that makes it easy for developers to create, publish, maintain, monitor, and secure APIs at any scale.

1) Go to API gateway

Click on Create API

Click on Rest API -> Build

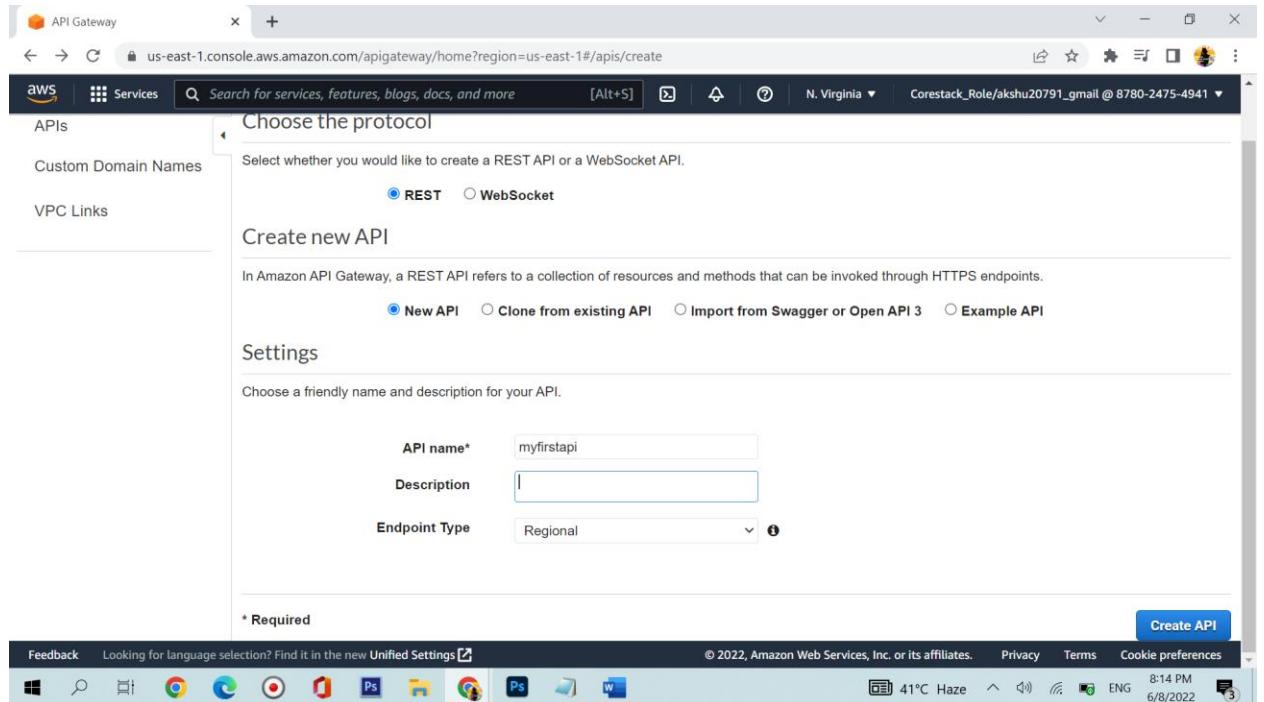


Now in Choose protocol -> Rest

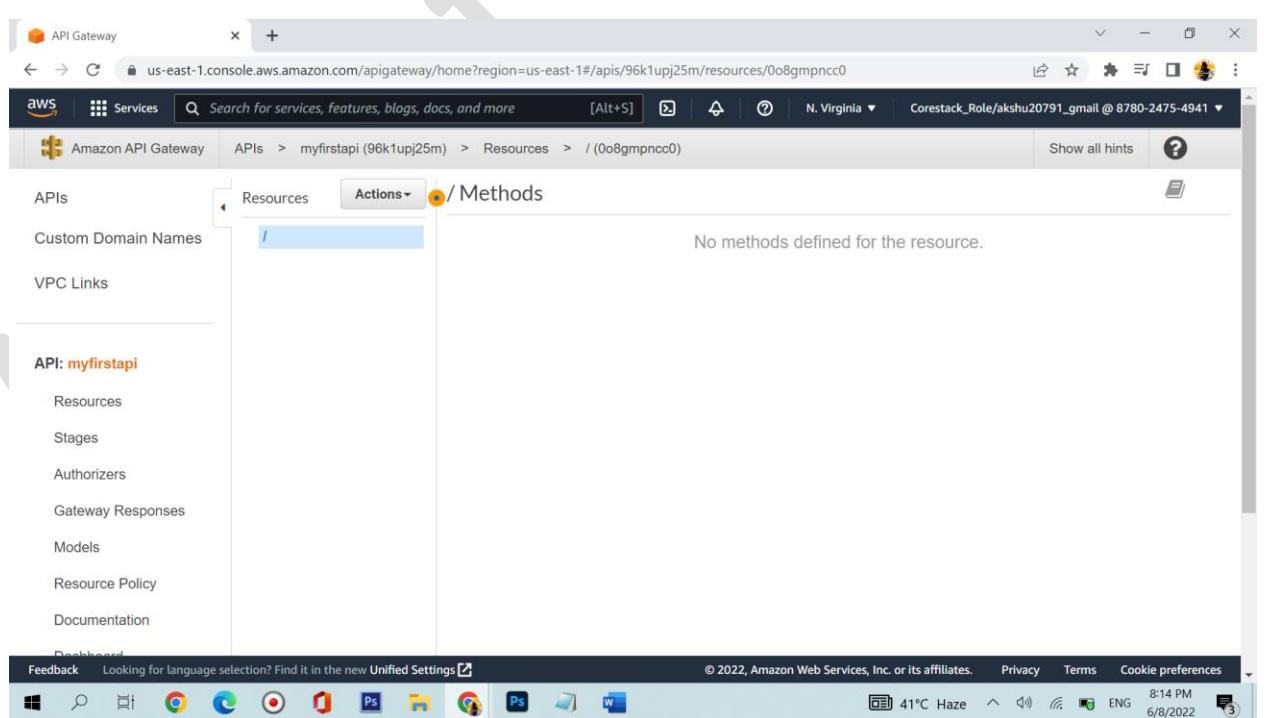
Create a new api -> New api

Api name : Myfirstapi

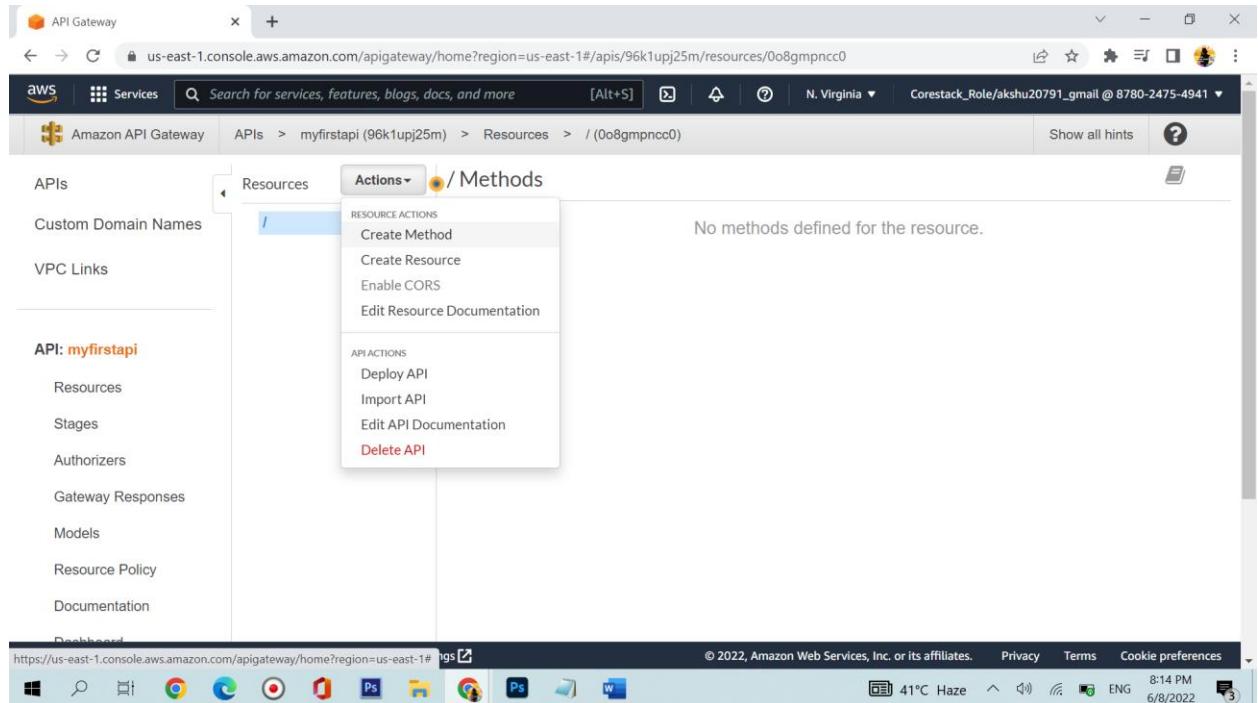
Endpoint-> regional



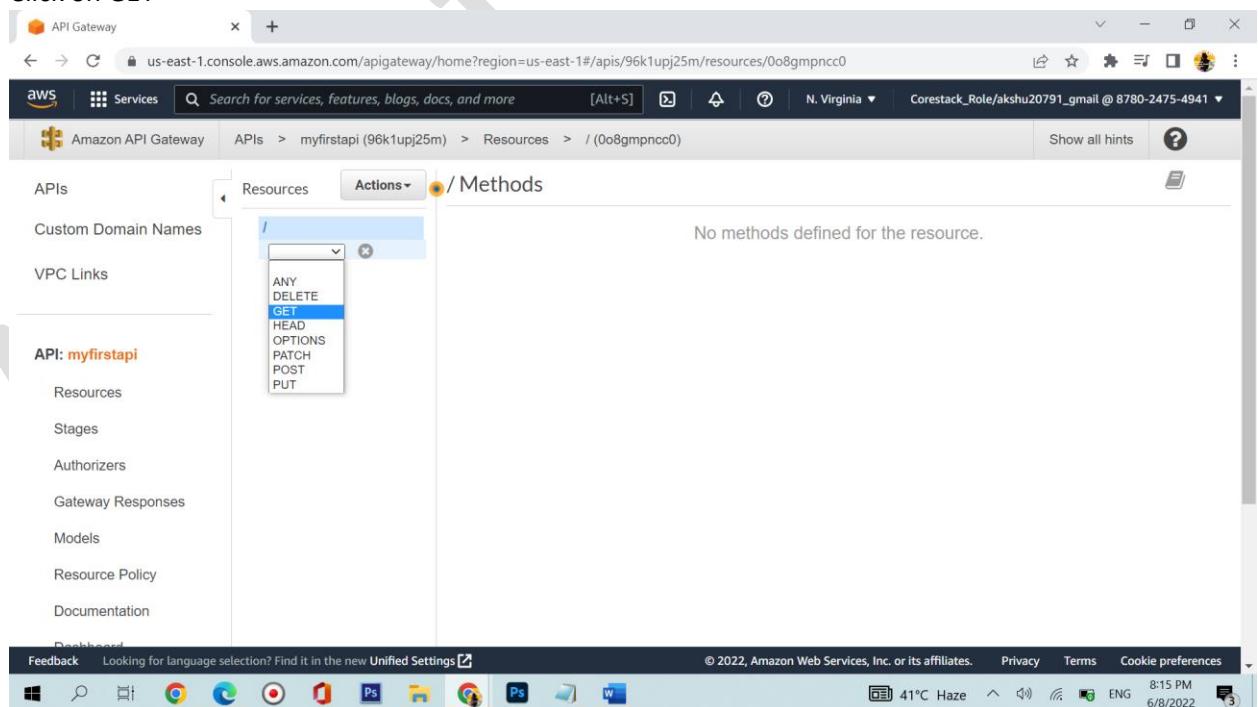
Such a page will open



Click on Actions -> Create method



Click on GET



The screenshot shows the AWS API Gateway setup interface. On the left, the navigation pane for 'API: myfirstapi' is visible, showing options like Resources, Stages, Authorizers, and Models. The main panel is titled '/ - GET - Setup' and asks 'Choose the integration point for your new method.' Below this, the 'Integration type' section is set to 'Lambda Function'. Other options include 'HTTP', 'Mock', 'AWS Service', and 'VPC Link'. A checkbox for 'Use Lambda Proxy integration' is checked. The 'Lambda Region' dropdown is set to 'us-east-1'. Under the 'Lambda Function' section, there is a text input field and a 'Use Default Timeout' checkbox, which is checked. At the bottom right of the panel is a 'Save' button.

We have to create a lambda function . Go to Lambda
Tick Use lambda proxy integration

Create function

The screenshot shows the AWS Lambda dashboard. The left sidebar includes 'Dashboard', 'Applications', 'Functions', 'Additional resources' (with 'Code signing configurations', 'Layers', and 'Replicas' listed), and 'Related AWS resources' (with 'Step Functions state machines' listed). The main area displays 'Resources for US East (N. Virginia)' with a summary table:

Lambda function(s)	Code storage	Full account concurrency	Unreserved account concurrency
2	611.0 byte (0% of 75.0 GB)	1000	1000

A 'Create function' button is located at the top right of this section. Below it is a 'Account-level metrics' section with three charts: 'Error count and success rate', 'Throttles', and 'Invocations'. The 'Error count and success rate' chart shows 1 error and 100 successes. The 'Throttles' chart shows 1 throttle. The 'Invocations' chart shows 4 invocations. At the bottom of the dashboard, there is a navigation bar with links for 'Settings', 'Privacy', 'Terms', and 'Cookie preferences', along with system status indicators like '41°C Haze' and '8:18 PM 6/8/2022'.

Function name: apigatewayfunctionforlambda

Runtime: Python 3.8

The screenshot shows the 'Basic information' section of the AWS Lambda function creation wizard. It includes fields for 'Function name' (apigatewayfunctionforlambda), 'Runtime' (Python 3.8), 'Architecture' (x86_64), and 'Permissions' (Change default execution role). The browser toolbar at the top indicates the URL is us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/create/function.

Create function

The screenshot shows the 'Function overview' page for the newly created function 'apigatewayfunctionforlambda'. It displays the function ARN (arn:aws:lambda:us-east-1:878024754941:function:apigatewayfunctionforlambda) and the Function URL (Info). The browser toolbar at the top indicates the URL is us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/functions/apigatewayfunctionforlambda?newFunction=true&tab=code.

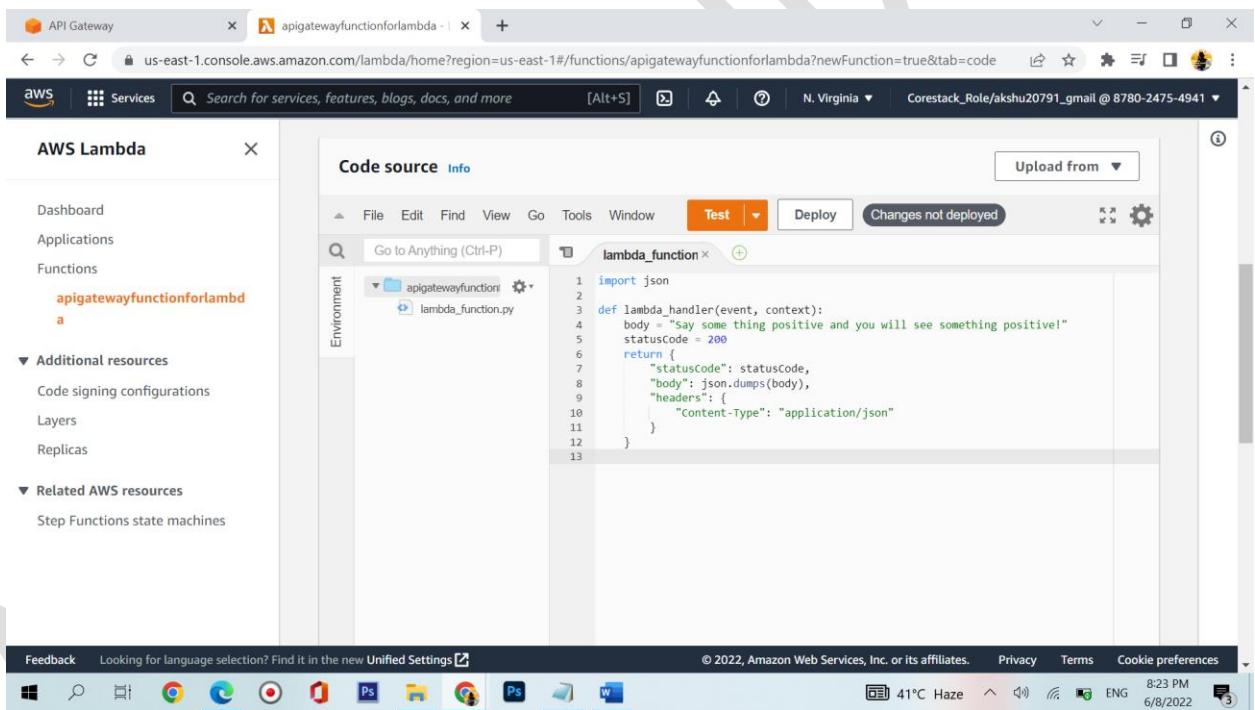
Now scroll down and paste the code

```

import json

def lambda_handler(event, context):
    body = "Say some thing positive and you will see something positive!"
    statusCode = 200
    return {
        "statusCode": statusCode,
        "body": json.dumps(body),
        "headers": {
            "Content-Type": "application/json"
        }
    }
}

```



Now click on test

The screenshot shows the AWS Lambda console interface. On the left, the navigation sidebar includes 'Dashboard', 'Applications', 'Functions' (with 'apigatewayfunctionforlambda' selected), 'Additional resources', 'Layers', 'Replicas', and 'Related AWS resources'. The main area is titled 'Code source' and contains the following Python code:

```
1 import json
2
3 def lambda_handler(event, context):
4     body = "Say something positive and you will see something positive!"
5     statusCode = 200
6     return {
7         "statusCode": statusCode,
8         "body": json.dumps(body),
9         "headers": {
10             "Content-Type": "application/json"
11         }
12     }
13
```

The screenshot shows the 'Test' configuration page for the 'apigatewayfunctionforlambda' function. The left sidebar is identical to the previous screenshot. The main area is titled 'Test event action' and shows a 'Create new event' button. Below it, the 'Event name' field contains 'test'. Under 'Event sharing settings', the 'Private' radio button is selected. In the 'Template - optional' section, 'hello-world' is selected. The 'Event JSON' section contains the following JSON:

```
1 {"key1": "value1", "key2": "value2"}
```

The screenshot shows the AWS Lambda function editor. On the left, the navigation bar includes 'Dashboard', 'Applications', 'Functions' (with 'apigatewayfunctionforlambda' selected), 'Additional resources', 'Related AWS resources', and 'Step Functions state machines'. The main area displays the following Python code:

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

On the right, there is a deployment preview window showing a simple JSON response with three key-value pairs. Below the code editor are 'Cancel' and 'Save' buttons.

Now deploy

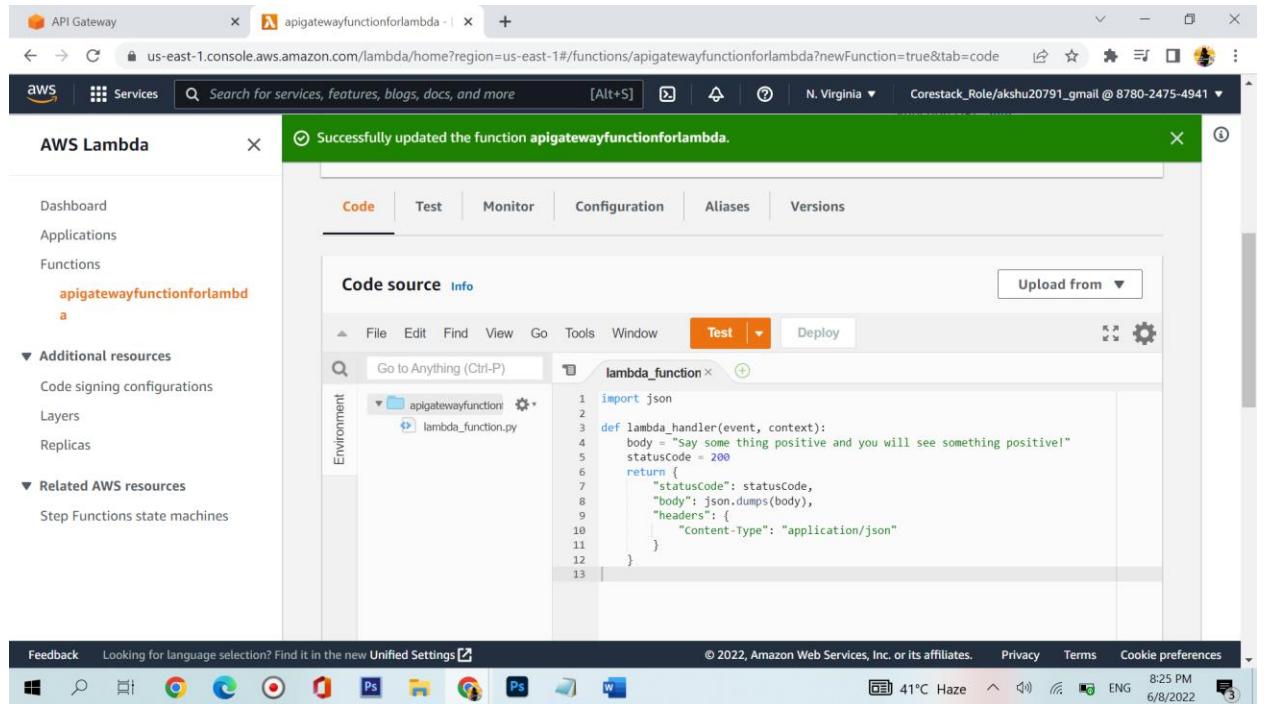
The screenshot shows the AWS Lambda function editor after deployment. A green success message at the top states: 'The test event test was successfully saved.' The navigation bar and sidebar are identical to the previous screenshot. The main area now shows the deployed code source in the 'Code source' tab:

```
Code source Info
```

The code is identical to the one shown in the first screenshot:1 import json
2
3 def lambda_handler(event, context):
4 body = "Say something positive and you will see something positive!"
5 statusCode = 200
6 return {
7 "statusCode": statusCode,
8 "body": json.dumps(body),
9 "headers": {
10 "Content-Type": "application/json"
11 }
12 }
13

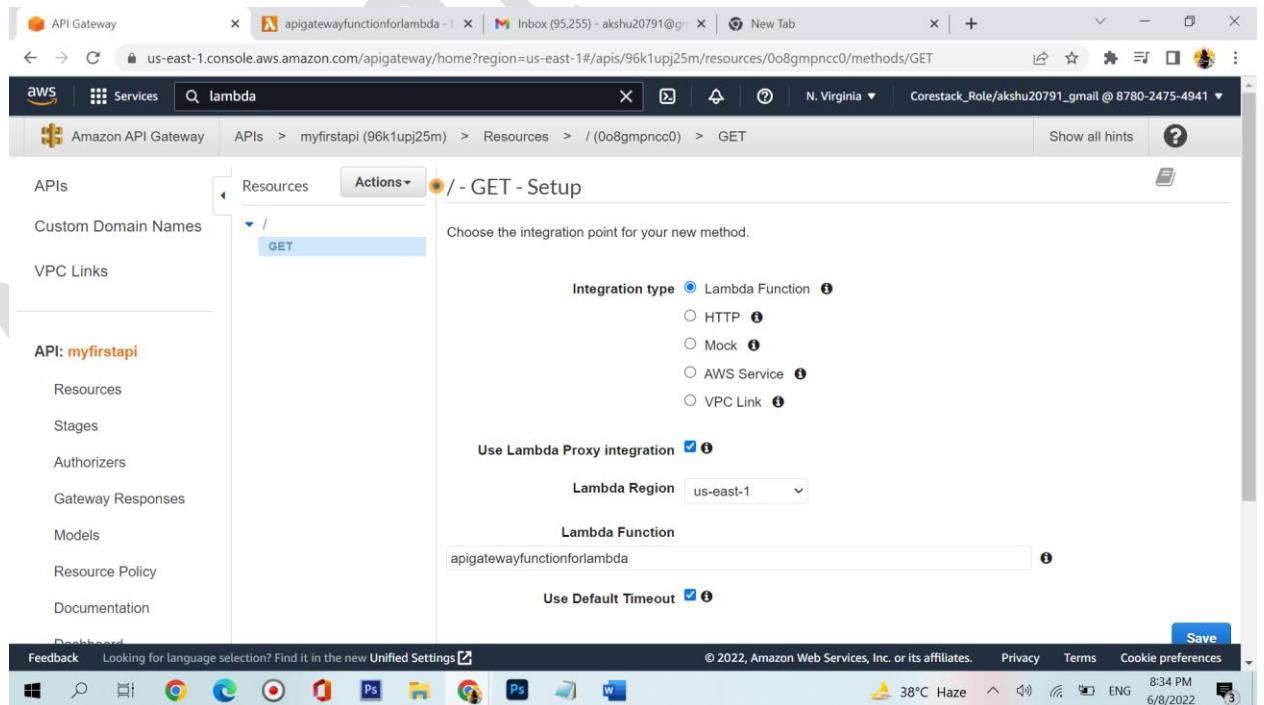
Below the code, there is a 'Test' button and a 'Deploy' button. The status bar indicates 'Changes not deployed'.

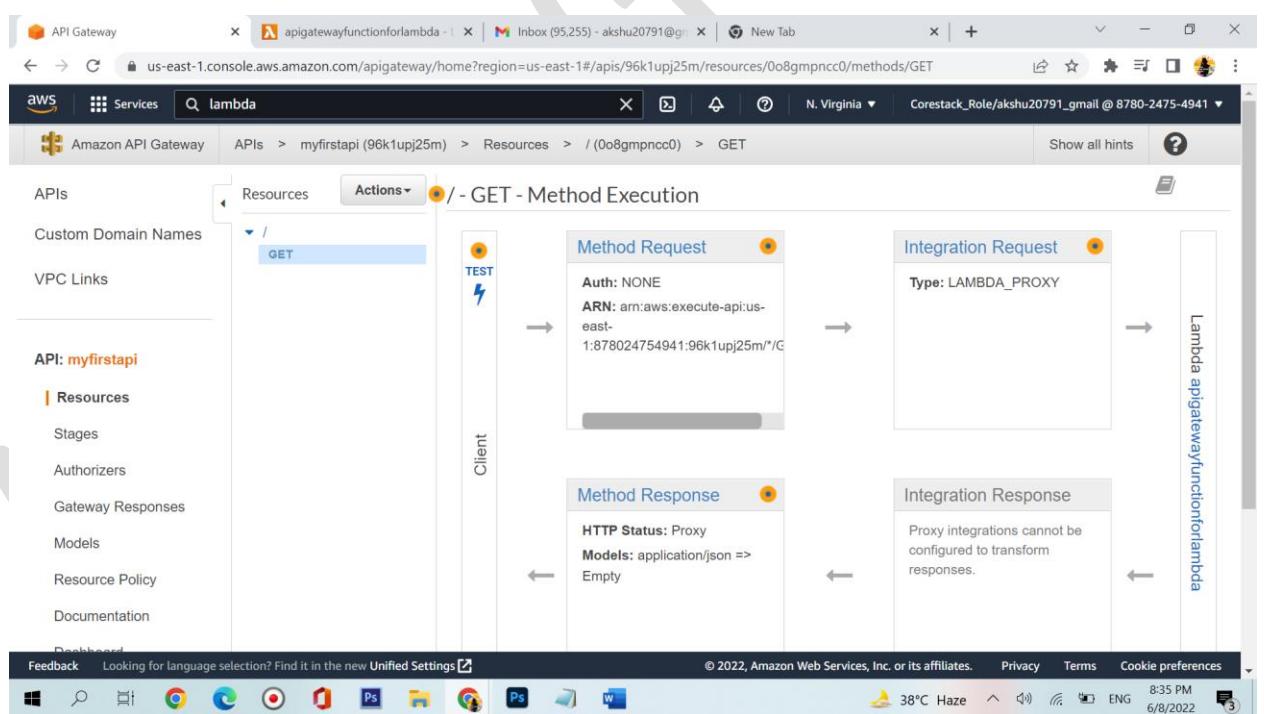
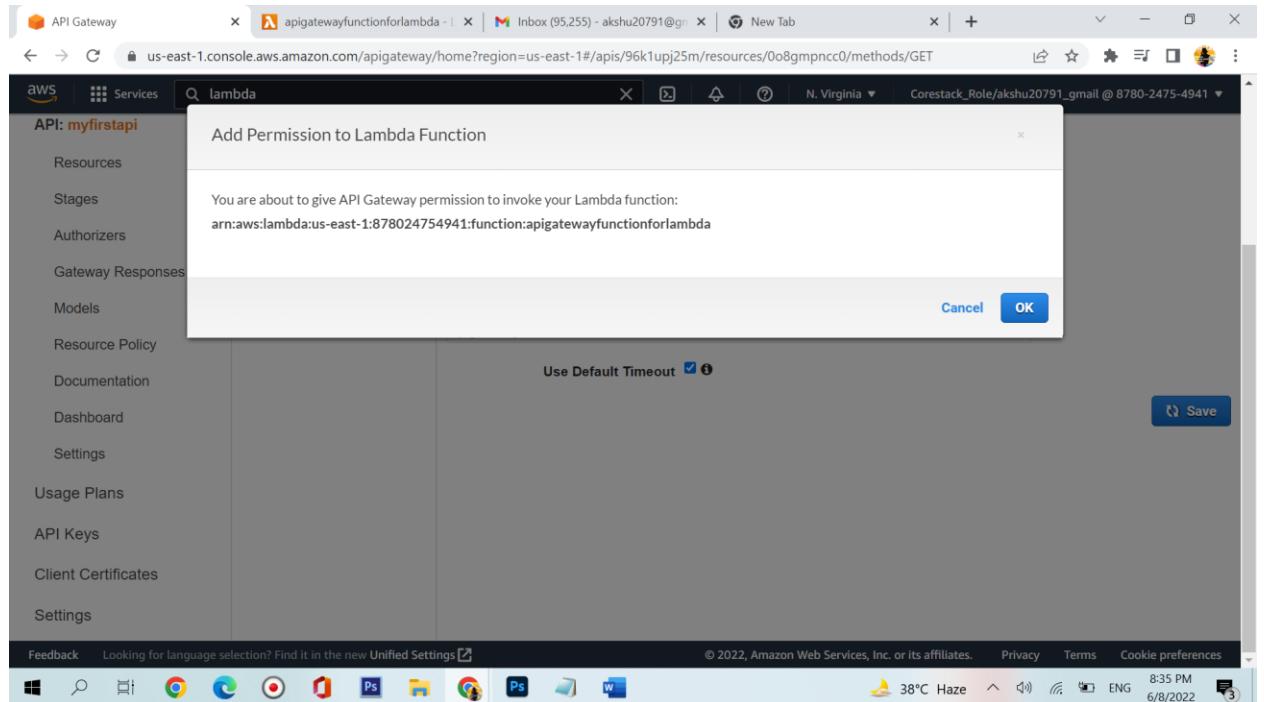
Changes deployed



Now copy the lambda function name and go back and paste it in API gateway

Paste apigatewayfunctionforlambda (lambda function name) in api gateway





Click on TEST

The screenshot shows the AWS Lambda API Gateway test interface. On the left, a sidebar lists options: Gateway Responses, Models, Resource Policy, Documentation, Dashboard, Settings, Usage Plans, API Keys, Client Certificates, and Settings. The main area contains several sections: Headers (No header parameters exist), Stage Variables (No stage variables exist), Client Certificate (No client certificates have been generated), Request Body (Request Body is not supported for GET methods), and a 'Test' button at the bottom.

Click test again

The screenshot shows the AWS Lambda API Gateway test interface after clicking the 'Test' button. The Response Headers section displays a JSON object: {"Content-Type": ["application/json"], "X-Amzn-Trace-Id": ["Root=1-62a0babca93413f4dde14477313dbc6; Sampled=0"]}. The Logs section shows an execution log for a request: Execution log for request afa8624c-094f-4d83-885a-d8dc7165459. The log details the execution flow from starting execution to the endpoint response.

Now Create resource

The screenshot shows the AWS API Gateway console. On the left, the navigation pane includes 'APIs', 'Custom Domain Names', 'VPC Links', and a section for 'API: myfirstapi' containing 'Resources', 'Stages', 'Authorizers', 'Gateway Responses', 'Models', 'Resource Policy', and 'Documentation'. The main area shows a 'Resources' list with a single item: '/ (0o8gmpncc0)'. A 'Actions' dropdown menu is open over this item, showing options like 'Edit Method Documentation', 'Delete Method', 'Create Method', 'Create Resource', 'Enable CORS', 'Edit Resource Documentation', 'Deploy API', 'Import API', and 'Edit API Documentation'. To the right of the resource list, there's a 'Method Execution / - GET - Method Test' panel. It displays a success message: 'Request: / Status: 200 Latency: 229 ms Response Body: "Say some thing positive and you will see something positive!"'. Below this, there's a 'Response Headers' section with JSON content and a 'Logs' section.

Give resource name

The screenshot shows the 'New Child Resource' creation interface. The left sidebar is identical to the previous screenshot. The main area has a title 'New Child Resource' with the sub-instruction 'Use this page to create a new child resource for your resource.' Below this, there's a 'Configure as proxy resource' section with a checked checkbox and a 'Resource Name*' field containing 'Staypositive'. There's also a 'Resource Path*' field with the value '/ staypositive'. A note explains that path parameters can be added using brackets. Below these fields are 'Enable API Gateway CORS' and 'Required' checkboxes, both of which are unchecked. At the bottom right are 'Cancel' and 'Create Resource' buttons.

The screenshot shows the AWS API Gateway console. On the left, the navigation pane for 'myfirstapi' is visible, with 'Resources' selected. In the main area, a 'New Child Resource' dialog is open. The path listed is '/'. Under 'Configure as proxy resource', the 'Resource Name' is set to 'Staypositive' and the 'Resource Path' is set to '/staypositive'. A note explains that you can add path parameters using brackets, such as '{username}' for a path parameter called 'username'. Below the dialog, there's a section for 'Enable API Gateway CORS' with a checkbox. At the bottom right of the dialog are 'Cancel' and 'Create Resource' buttons.

Create method

The screenshot shows the AWS API Gateway console. The path in the navigation bar is 'myfirstapi /staypositive'. A context menu is open over the '/staypositive' resource, with the 'Actions' tab selected. The menu items under 'RESOURCE ACTIONS' include 'Create Method', 'Create Resource', 'Enable CORS', 'Edit Resource Documentation', and 'Delete Resource'. Under 'API ACTIONS', the options are 'Deploy API', 'Import API', 'Edit API Documentation', and 'Delete API'. To the right of the menu, a message states 'No methods defined for the resource.'

Select GET

The screenshot shows the AWS API Gateway console. In the top navigation bar, there are tabs for 'API Gateway', 'apigatewayfunctionforlambda', 'Inbox (95,255) - akshu20791@gmail.com', and 'New Tab'. Below the navigation bar, the URL is 'us-east-1.console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/96k1upj25m/resources/v7k9nh'. The main content area shows an 'Amazon API Gateway' interface. On the left, a sidebar lists 'APIs', 'Custom Domain Names', and 'VPC Links'. Under 'APIs', 'myfirstapi' is selected. The main panel shows 'Resources' and 'Actions'. A 'GET' method for the path '/staypositive' is listed, with its details shown in a modal window. The modal shows the method type 'GET', a dropdown menu, and a checkmark icon. The status message 'No methods defined for the resource.' is displayed below the modal.

Click on Tick

This screenshot is identical to the one above, showing the AWS API Gateway console with the same interface and the same GET method for the '/staypositive' endpoint. The difference is that the checkmark icon in the modal window for the GET method is now checked, indicating that the lambda proxy integration has been enabled.

Tick use lambda proxy integration

The screenshot shows the AWS API Gateway Lambda integration configuration. On the left, the sidebar lists the API named "myfirstapi" and its resources like Stages, Authorizers, and Models. The main panel shows a method mapping for the path "/staypositive" with a GET operation. The "Integration type" is set to "Lambda Function". The "Lambda Region" is "us-east-1". The "Lambda Function" field is empty. The "Use Default Timeout" checkbox is checked. A "Save" button is visible at the bottom right.

For lambda function ..again go to lambda function

Create a new lambda function

The screenshot shows the AWS Lambda console. The left sidebar includes options like Dashboard, Applications, Functions, and Additional resources. The main area displays "Resources for US East (N. Virginia)" with a summary table. It shows 3 Lambda functions, 932.0 byte of code storage, 1000 full account concurrency, and 1000 unreserved account concurrency. A "Create function" button is at the top right. Below this is a section for "Account-level metrics" with charts for Error count and success rate, Throttles, and Invocations. The time range for the metrics is set to 1h. The status bar at the bottom indicates the URL is https://us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1..., the date is 6/8/2022, and the time is 8:38 PM.

Give name : staypositivelambdafunction

Runtime: python 3.8

The screenshot shows the 'Basic information' step of the AWS Lambda function creation wizard. The 'Function name' field is filled with 'staypositivelambdafunction'. The 'Runtime' dropdown is set to 'Python 3.8'. The 'Architecture' dropdown shows 'x86_64' selected. Under 'Permissions', it says 'By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.' A link '▶ Change default execution role' is visible.

The screenshot shows the 'Functions' page in the AWS Lambda console. It displays the newly created function 'staypositivelambdafunction'. The 'Function overview' section shows the function icon, name, and a note that it was last modified 8 seconds ago. It also lists the Function ARN and Function URL. A green banner at the top states: 'Successfully created the function staypositivelambdafunction. You can now change its code and configuration. To invoke your function with a test event, choose "Test".'

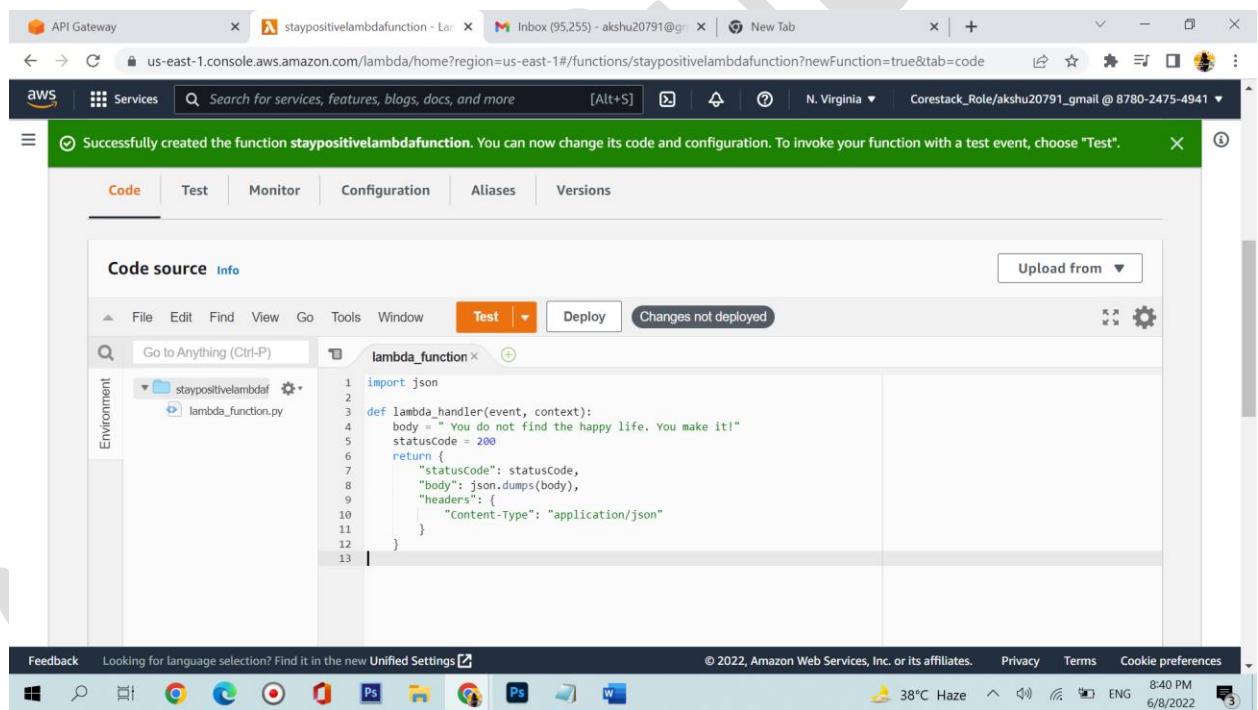
Put the code

```

import json

def lambda_handler(event, context):
    body = " You do not find the happy life. You make it!"
    statusCode = 200
    return {
        "statusCode": statusCode,
        "body": json.dumps(body),
        "headers": {
            "Content-Type": "application/json"
        }
    }
}

```



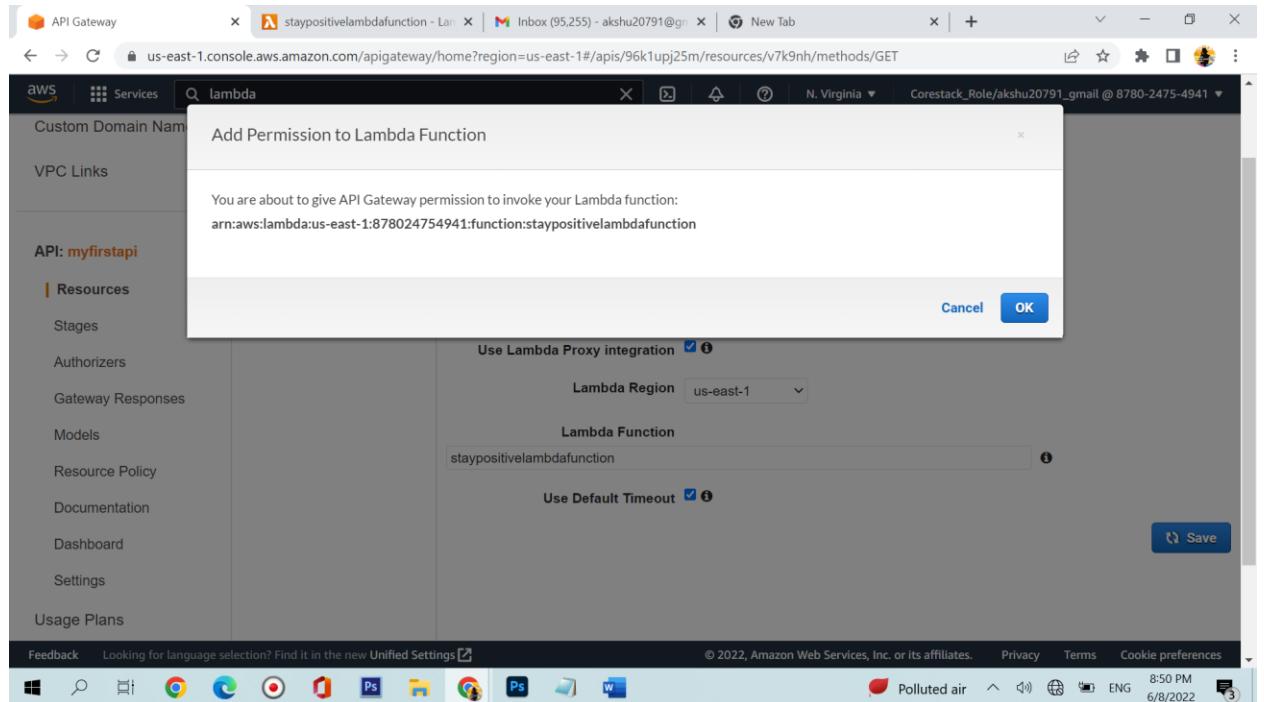
Deploy the code

The screenshot shows the AWS Lambda function editor. At the top, there are tabs for Code, Test, Monitor, Configuration, Aliases, and Versions. The Code tab is selected. Below the tabs, there is a toolbar with File, Edit, Find, View, Go, Tools, Window, Test, Deploy, and a dropdown for Upload from. The main area is titled "Code source Info" and contains a code editor for "lambda_function.py". The code is as follows:

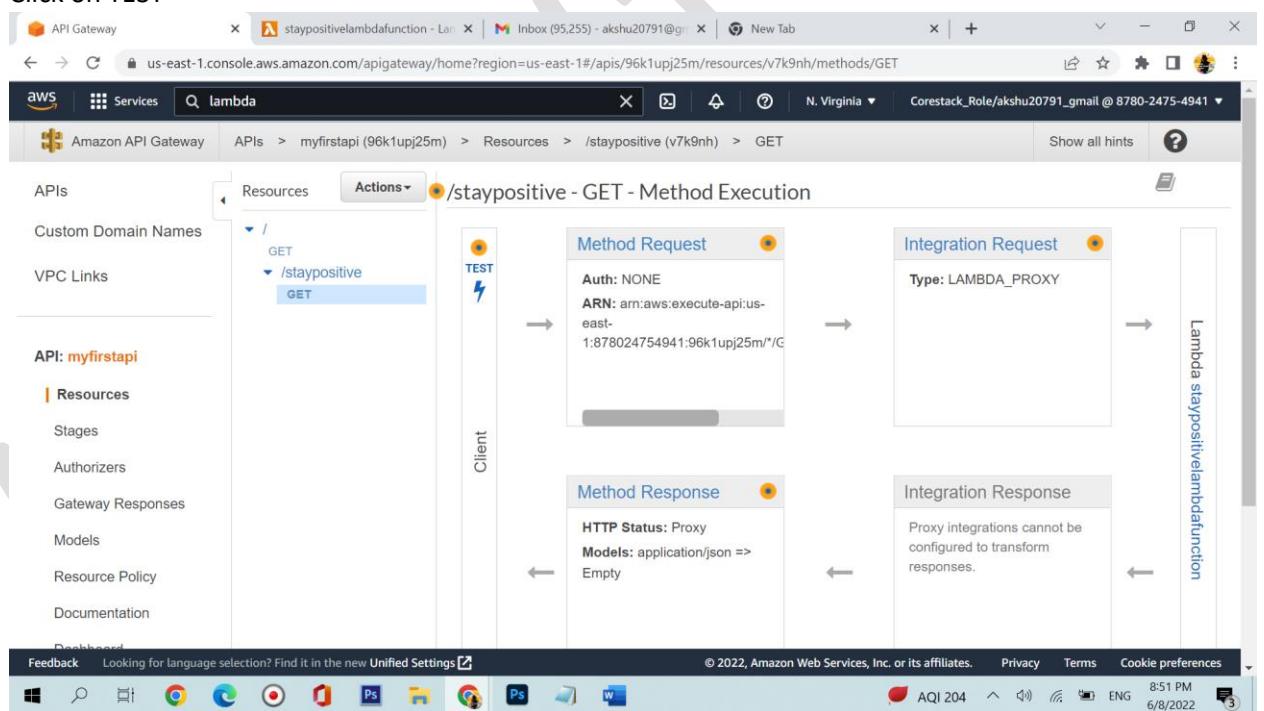
```
1 import json
2
3 def lambda_handler(event, context):
4     body = "You do not find the happy life. You make it!"
5     statuscode = 200
6     return {
7         "statusCode": statusCode,
8         "body": json.dumps(body),
9         "headers": {
10             "Content-Type": "application/json"
11         }
12     }
13
```

Come back to API gateway and paste the lambda function name there

The screenshot shows the AWS API Gateway integration configuration screen. On the left, there is a sidebar with options: Custom Domain Names, VPC Links, API: myfirstapi, Resources, Stages, Authorizers, Gateway Responses, Models, Resource Policy, Documentation, Dashboard, and Settings. The Resources section is expanded, showing a list of resources: / (GET), /staypositive (GET), and /v7k9nh (METHODS/GET). The /staypositive resource is selected. The main panel shows the integration configuration for this resource. It includes fields for Integration type (set to Lambda Function), Use Lambda Proxy integration (checked), Lambda Region (set to us-east-1), Lambda Function (set to staypositivelambdafunction), and Use Default Timeout (checked). A Save button is located at the bottom right.



Click on TEST



Click on TEST

The screenshot shows the AWS Lambda API Gateway configuration interface. On the left, a sidebar lists various options: Models, Resource Policy, Documentation, Dashboard, Settings, Usage Plans, API Keys, Client Certificates, and Settings. The main content area is titled 'Headers' and contains a section for 'staypositive'. It includes a note: 'Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg. Accept:application/json.' Below this are sections for 'Stage Variables', 'Client Certificate', and 'Request Body', each with descriptive text. At the bottom right is a blue 'Test' button.

The screenshot shows the AWS Lambda API Gateway test interface. The top navigation bar shows the path: APIs > myfirstapi (96k1upj25m) > Resources > /staypositive (v7k9nh) > GET. The main area is titled 'Method Execution /staypositive - GET - Method Test'. On the left, a sidebar shows the API structure: APIs, Custom Domain Names, VPC Links, and the selected API 'myfirstapi'. Under 'Resources', it shows a tree structure with a root node 'GET' expanded to show '/staypositive' and another 'GET' node. The right side displays the test results: 'Path' (Request: /staypositive, Status: 200, Latency: 240 ms), 'Query Strings' (param1=value1¶m2=value2), 'Headers' (staypositive), and 'Response Body' (You do not find the happy life. You make it!). Below these is a 'Response Headers' section showing Content-Type: application/json, X-Amzn-Trace-Id: Root=1-62abbe9d-0a49005db9cc355948dc0df0; Sampled=0, and a 'Logs' section with the execution log: Execution log for request 45fc90c0-0229-401f-a422-.

Now click on Actions -> Deploy API

API: myfirstapi

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Deployment

Dashboard

Actions

/

GET

/staypositive

GET

METHOD ACTIONS

Edit Method Documentation

Delete Method

RESOURCE ACTIONS

Create Method

Create Resource

Enable CORS

Edit Resource Documentation

Delete Resource

API ACTIONS

Deploy API

Import API

Edit API Documentation

Delete API

Request: /staypositive

Status: 200

Latency: 240 ms

Response Body

You do not find the happy life. You make it!

Response Headers

{"Content-Type": ["application/json"], "X-Amzn-Trace-Id": ["Root=1-62a0be9d-0a49005db9cc355948dc0df0; Sampled=0"]}

Logs

Execution log for request 45fc90c0-0229-401f-a422-

Deployment stage -> newstage

Stagename: teststage

Deploy

API: myfirstapi

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Deployment

Feedback

Looking for language selection? Find it in the new Unified Settings

Services

lambda

APIs > myfirstapi (96k1upj25m) > Resources > /staypositive (v7k9nh) > GET

Actions

/

GET

/staypositive

GET

Deployment stage

[New Stage]

Stage name*

teststage

Stage description

Deployment description

Cancel Deploy

Request: /staypositive

Status: 200

Latency: 240 ms

Response Body

You do not find the happy life. You make it!

Response Headers

{"Content-Type": ["application/json"], "X-Amzn-Trace-Id": ["Root=1-62a0be9d-0a49005db9cc355948dc0df0; Sampled=0"]}

Logs

Execution log for request 45fc90c0-0229-401f-a422-

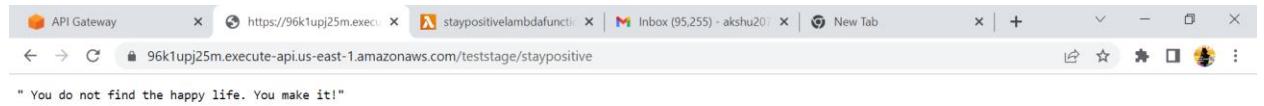
The screenshot shows the AWS API Gateway Stage Editor. On the left, a sidebar lists the API: 'myfirstapi' with sections for Resources, Stages (selected), Authorizers, Gateway Responses, Models, Resource Policy, Documentation, and Dashboard. The main area is titled 'teststage Stage Editor'. It displays the 'Invoke URL' as <https://96k1upj25m.execute-api.us-east-1.amazonaws.com/teststage>. Below this are tabs for Settings, Logs/Tracing, Stage Variables, SDK Generation, Export, and Deployment History. Under Settings, there's a 'Cache Settings' section with an 'Enable API cache' checkbox (unchecked). The 'Default Method Throttling' section indicates a rate of 10000 requests per second. A note states: 'Choose the default throttling level for the methods in this stage. Each method in this stage will respect these rate and burst settings. Your current account level throttling rate is 10000 requests per second with a burst of 5000 requests.' A link to 'Read more about API Gateway throttling' is provided. At the bottom, there are links for Documentation History and Canary.

Click on invoke url

The screenshot shows a browser window with the URL <https://96k1upj25m.execute-api.us-east-1.amazonaws.com/teststage>. The page content is a single line of text: "Say some thing positive and you will see something positive!"

Add /staypositive in url

<https://96k1upj25m.execute-api.us-east-1.amazonaws.com/teststage/staypositive>



#####THANKYOU #####