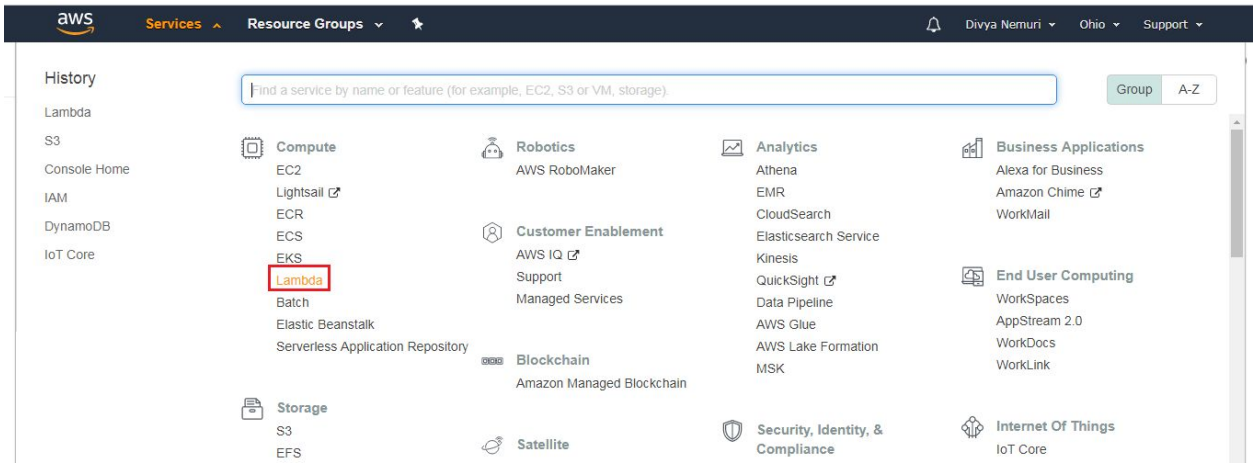
**Aim:** To create an API to publish the data to AWS IOT core from the mobile application using the HTTP request.

**Services Used:**

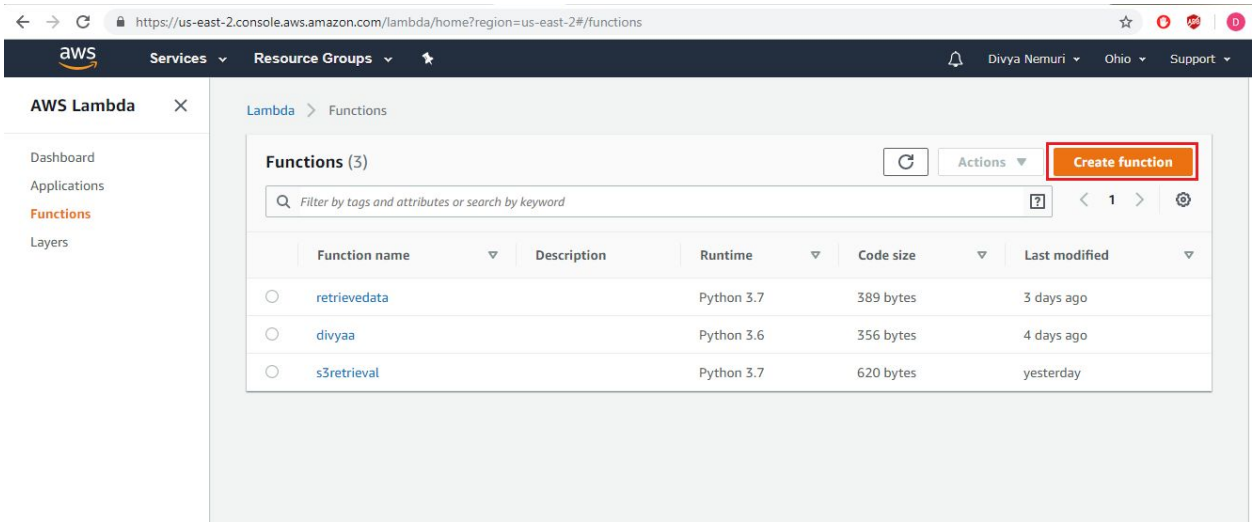
- AWS IoT Core
- Lambda
- API gateway
- IAM Service

Create a lambda function

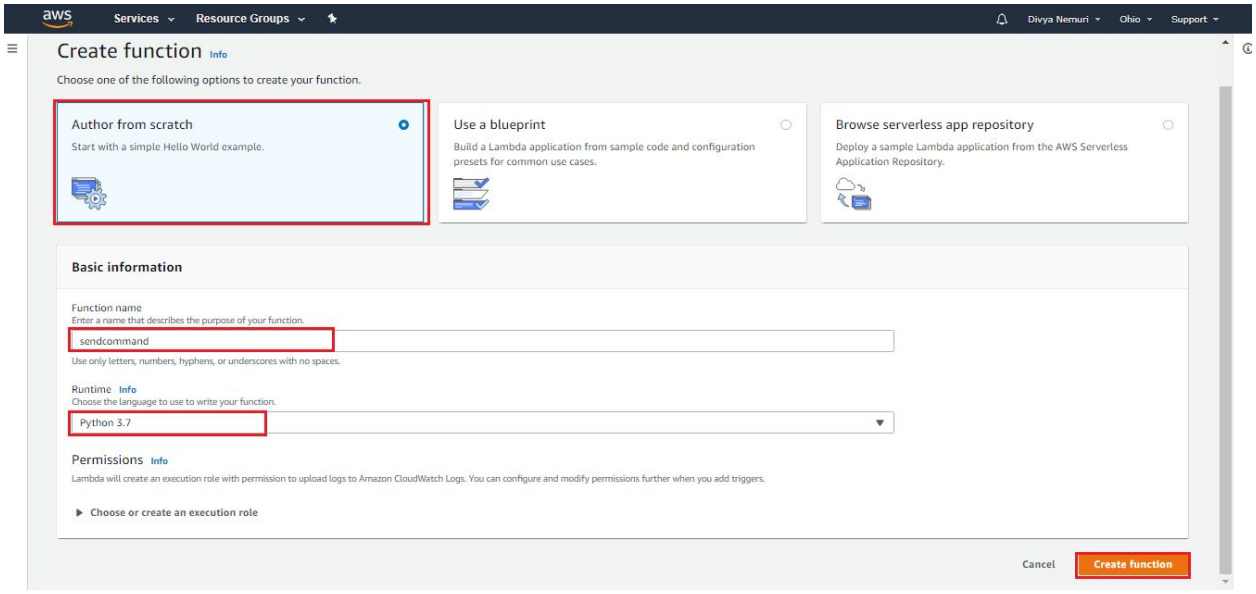
- Login to your AWS console and search for lambda service



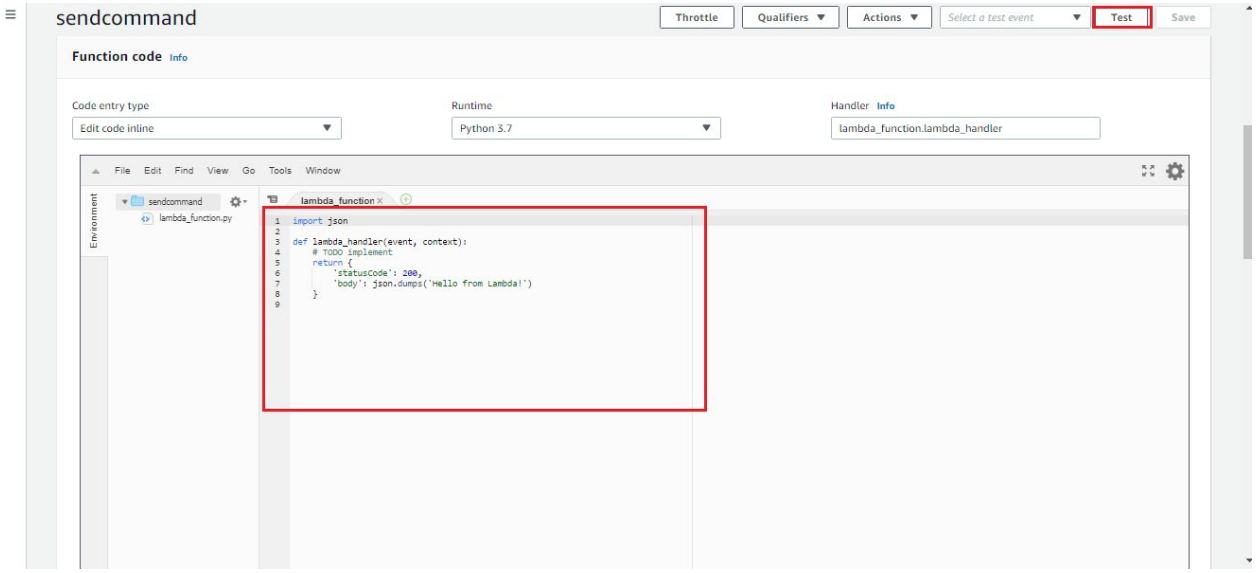

- **Double click on the lambda service and click on create function**



- There you select an author from scratch, give some function name and select the language as python 3.7



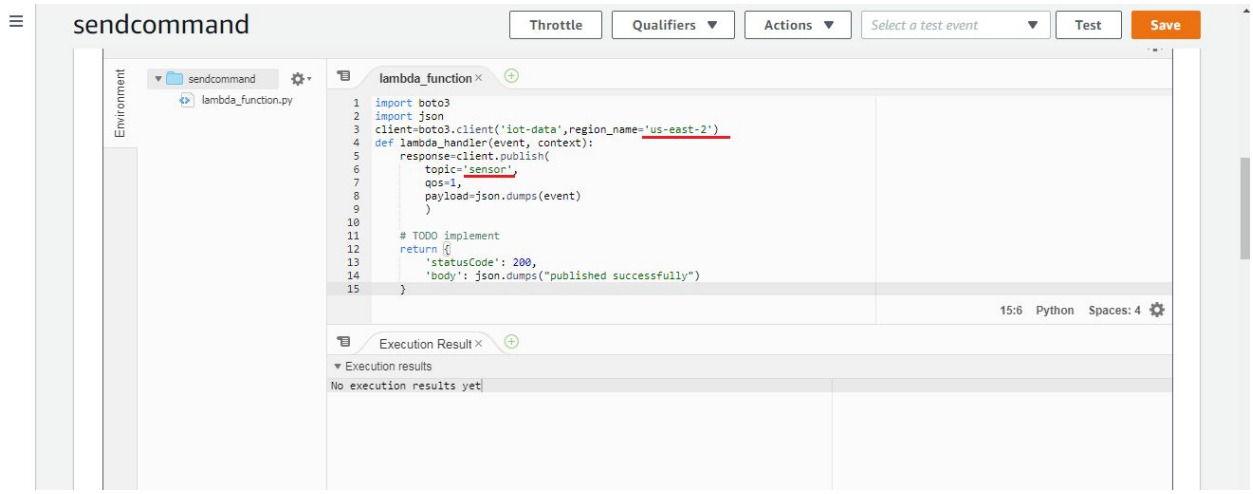

- After selecting the required parameters click on create function. then you will be getting a code editor. there you write your python code for publishing data to the AWS IOT topic.



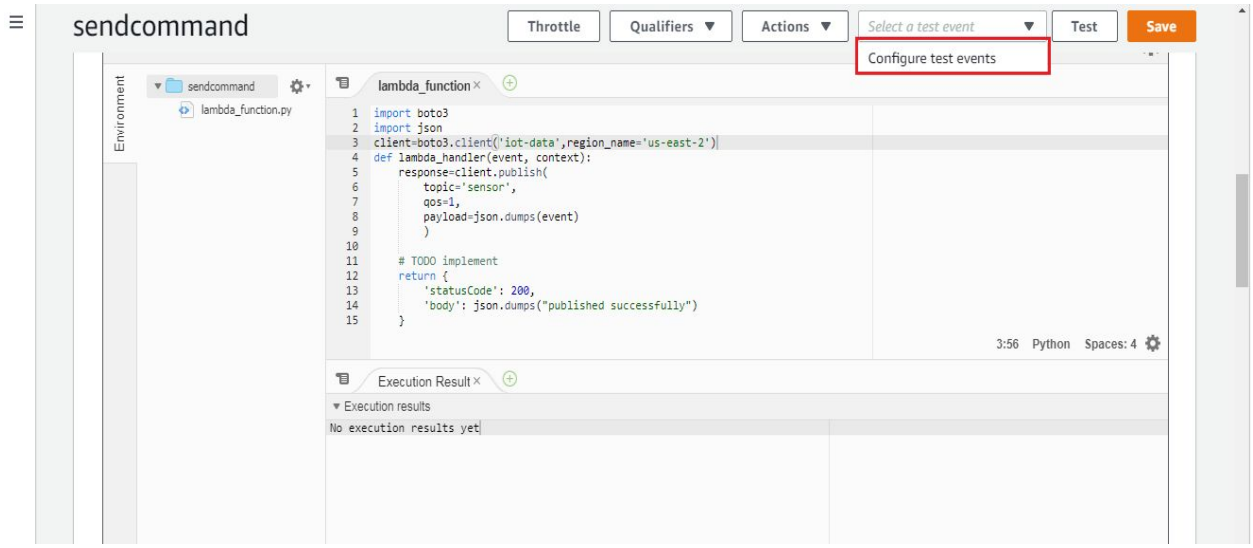

- In the code change the topic name and the aws IoT region in which you have created the AWS IOT service.

**Lambda function-python code:**

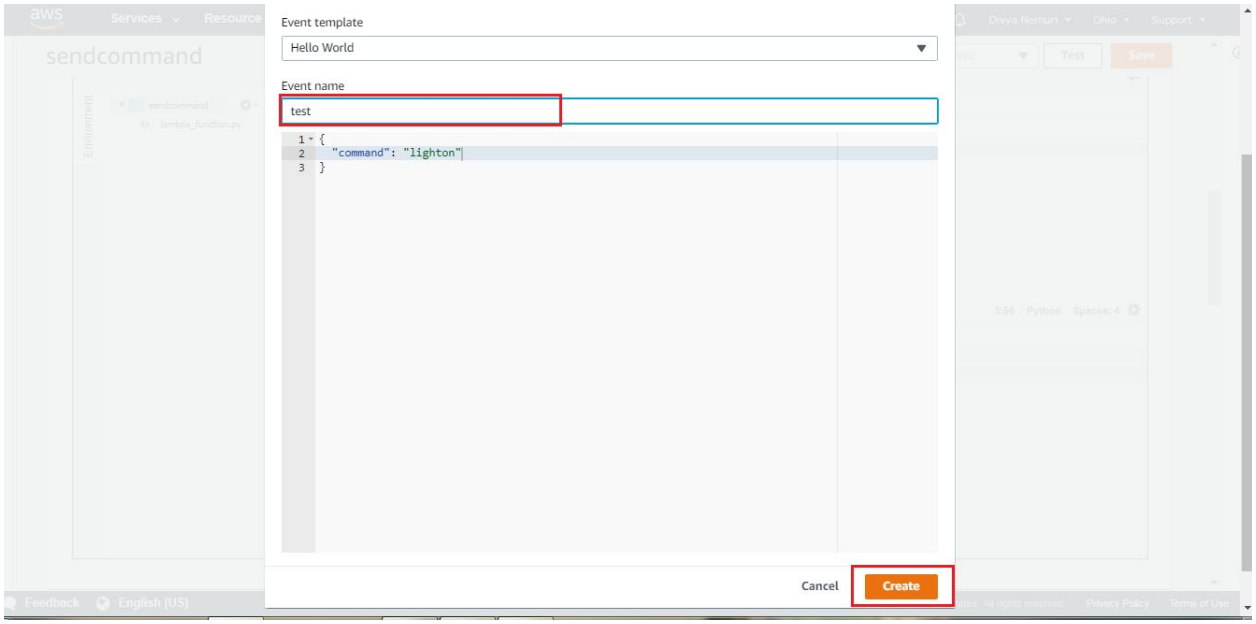
```
import boto3
import json
client=boto3.client('iot-data',region_name='us-east-2')
def lambda_handler(event, context):
    response=client.publish(
        topic='sensor',
        qos=1,
        payload=json.dumps(event)
        )

    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps("published successfully")
    }
```



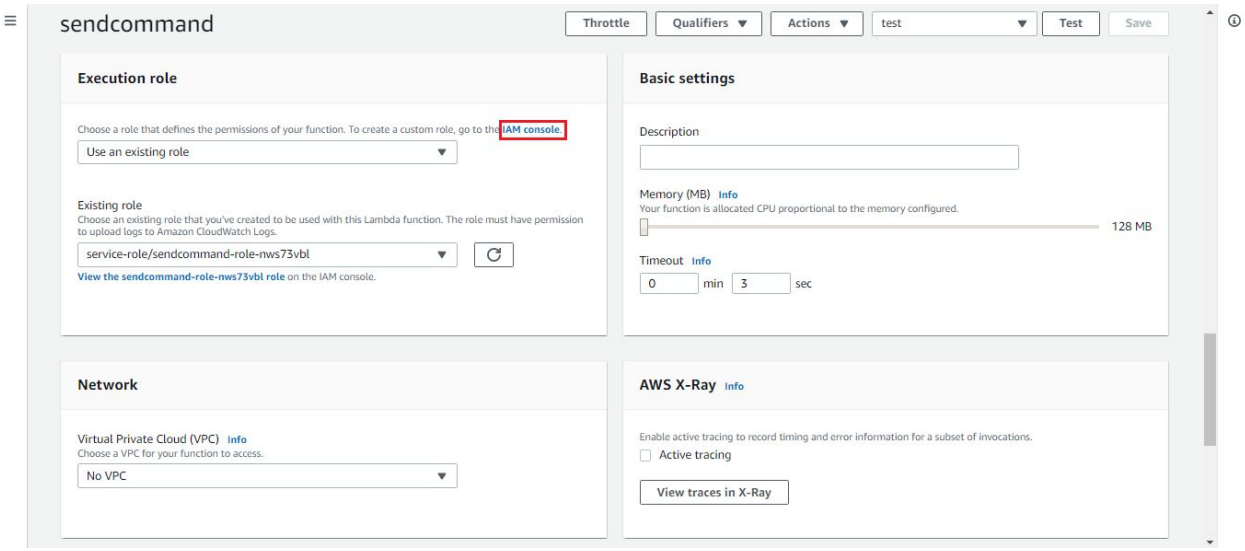

- Now in order to test your Lambda function whether it is correct or not, you can test it by using the test option.
- As we don't have any API now we can pass the data using the configure test events.



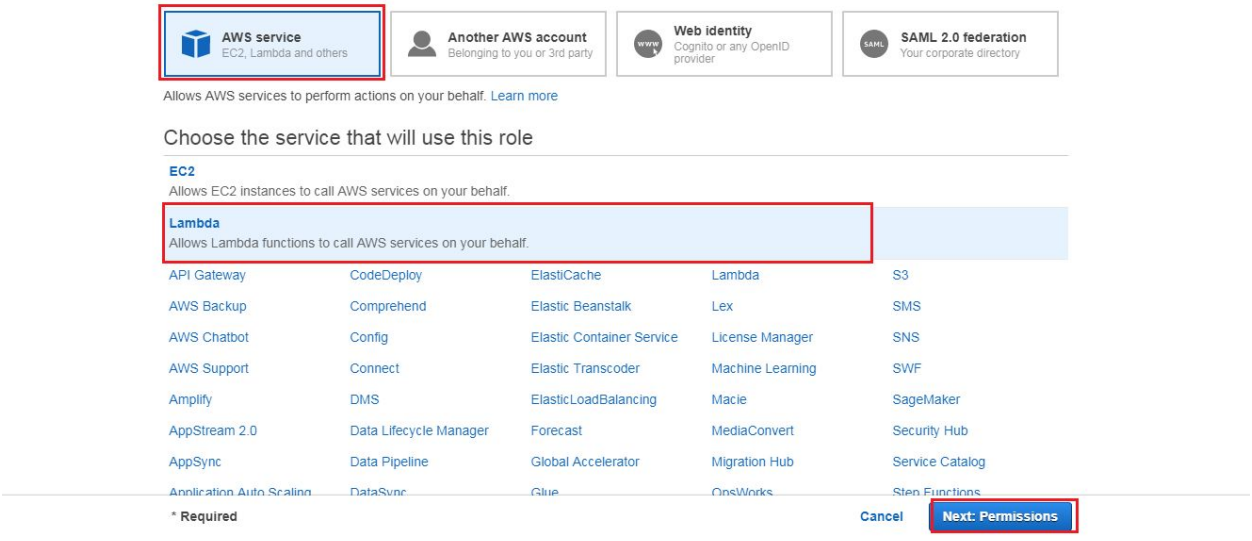

- Click on the configure test event and there enter some name.
- Enter the JSON data {“command”:”lighton”} or {“message”:”lighton”} and click on create.



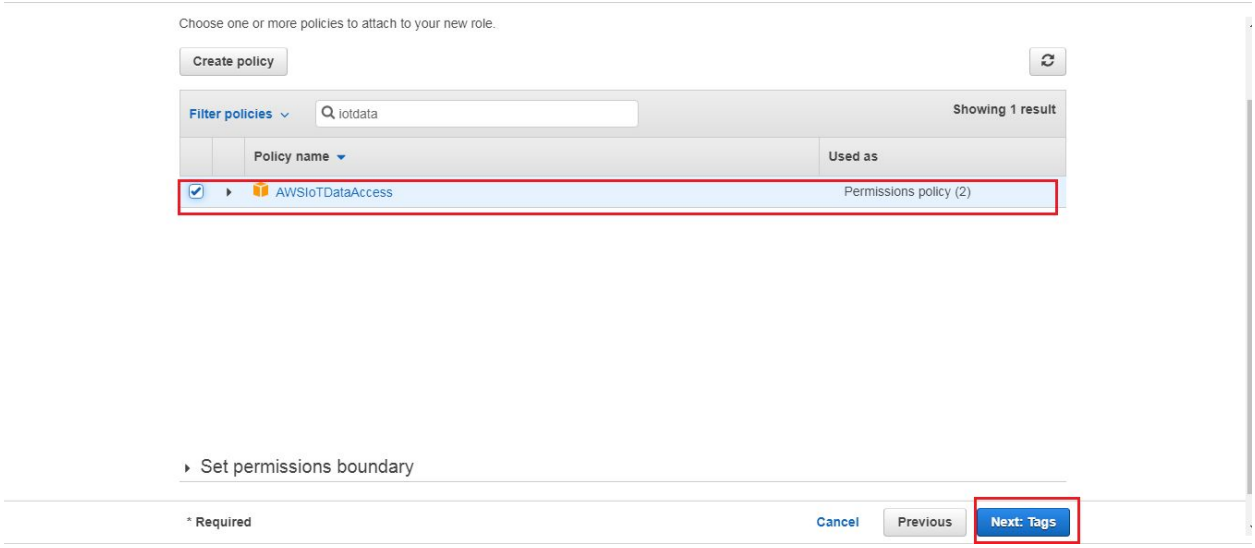

- Now to test your Lambda function we need to attach the IAM role for accessing the AWS IOT core using the lambda function.
- To attach the IAM role just scroll down and under the execution role, you can select your IAM role.



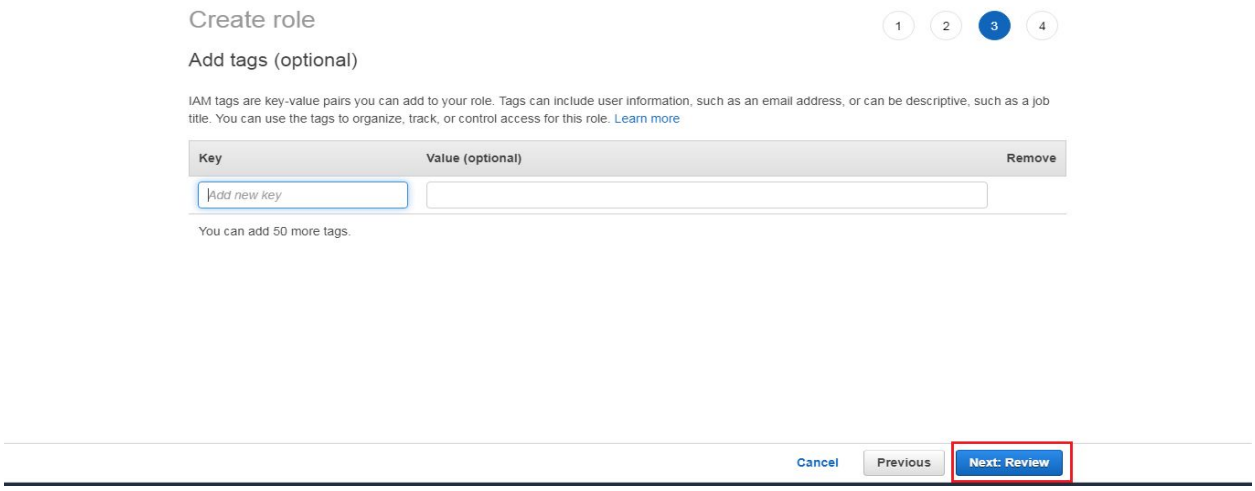

- Under the execution role click on the IAM console, you will be redirected to the IAM console where you can create the IAM role.
- In that console select the AWS Service and choose the service as lambda and click on next.



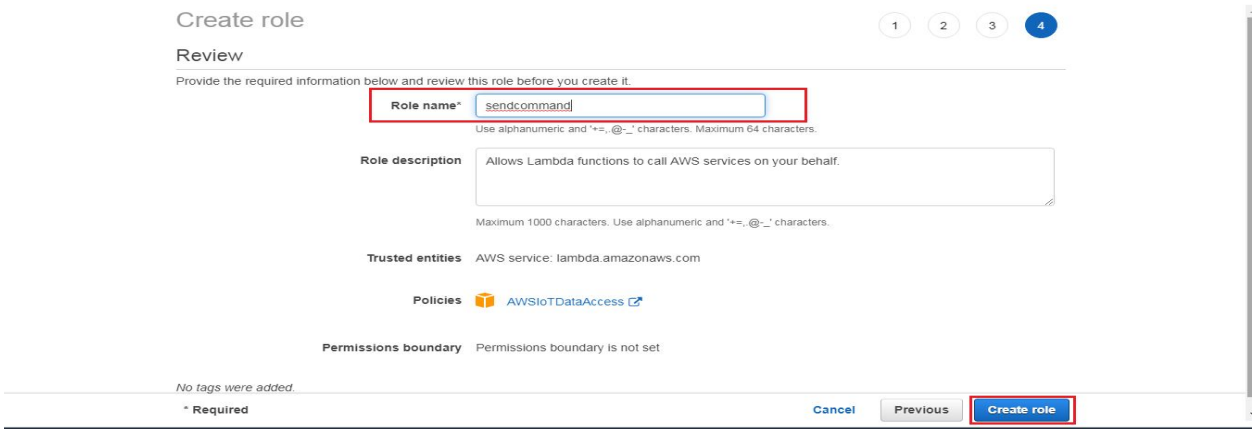

- In the next page search for AWSIOTDATAACCESS, select that policy and click on next.



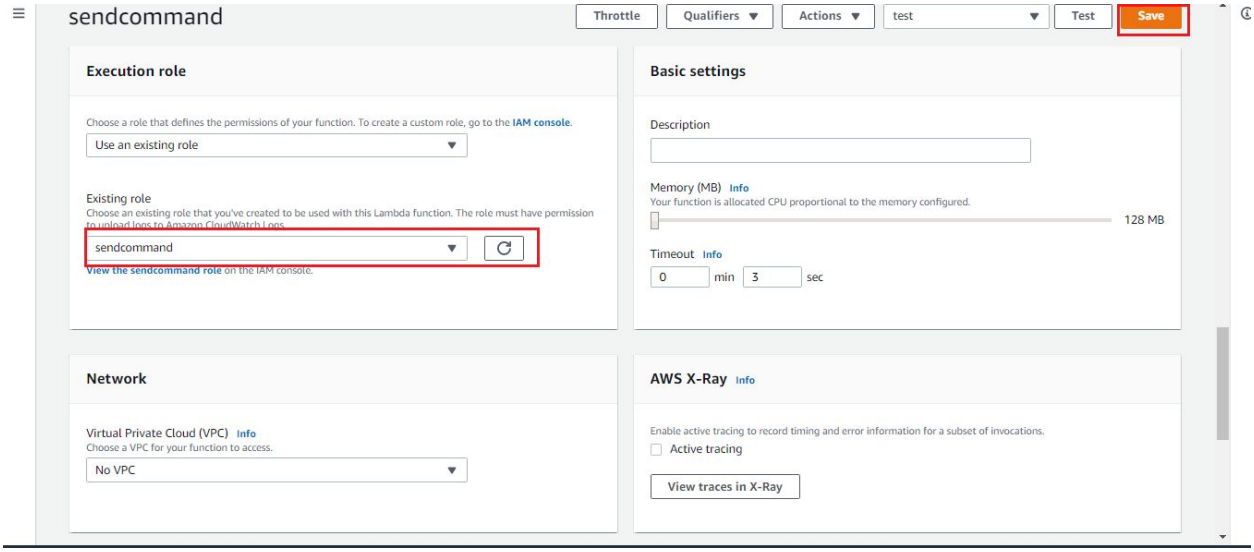

- On the next page leave the key value as empty and click on next.



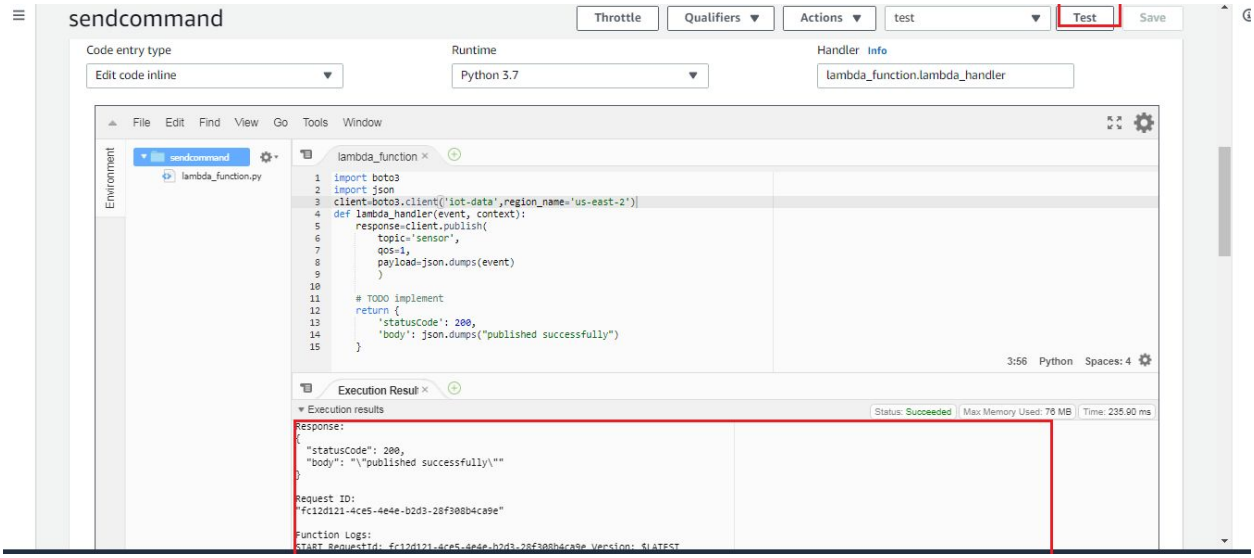

- Here it will ask to enter the role name , give some name and click on create role.



- Now go back to your lambda page and under the execution role select your IAM role which you have created. If you don't get your role in the existing list just refresh that and select the desired role.

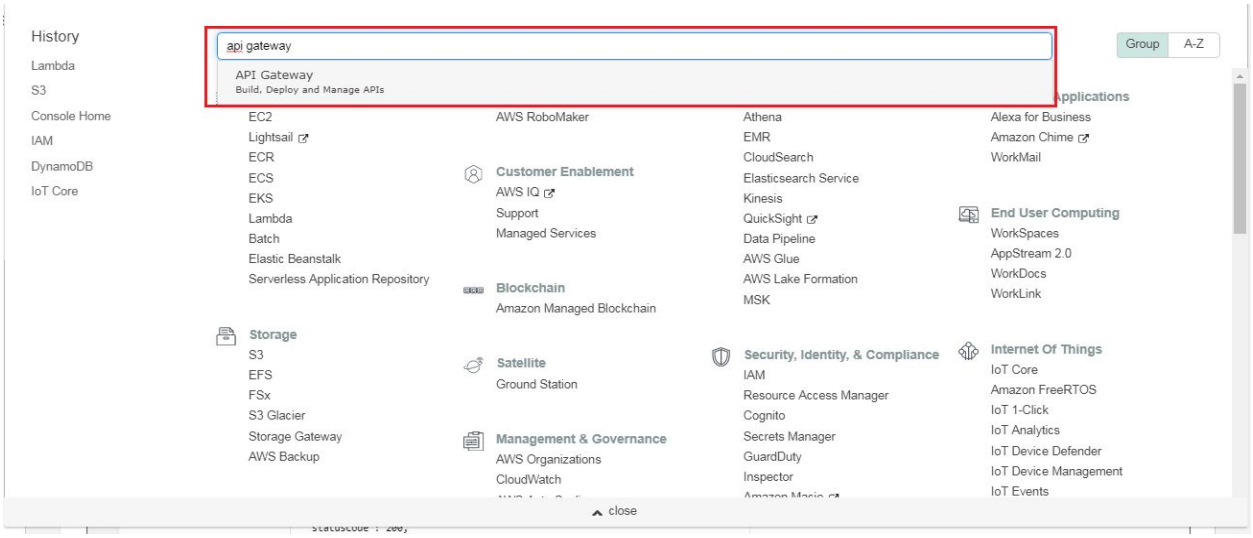



- After selecting the IAM role you can save your lambda function and now you can test it by clicking on the test option.

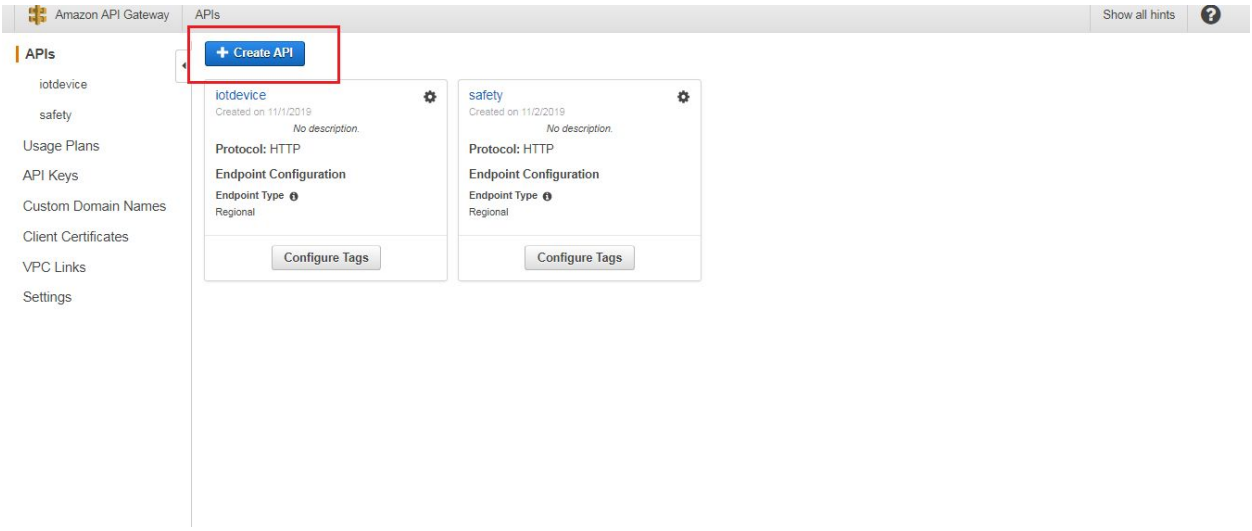



- After testing you can check the execution results. You need to get published successfully message.
- If you got that message then your lambda function is created successfully and then you need to create an API by using the API gateway.

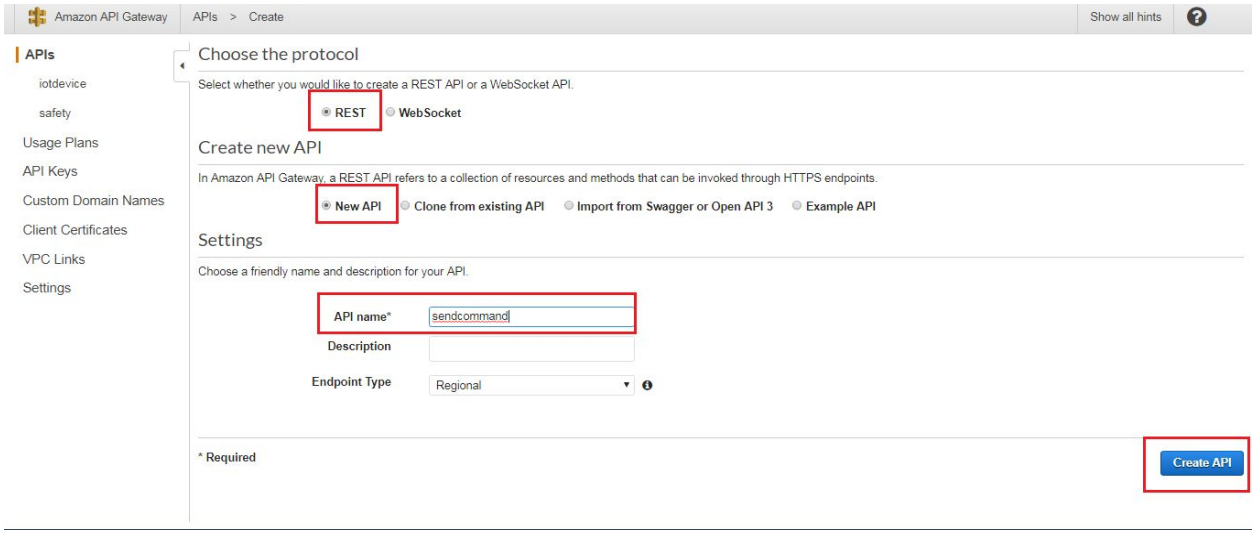
- Now you search for API gateway under the services.



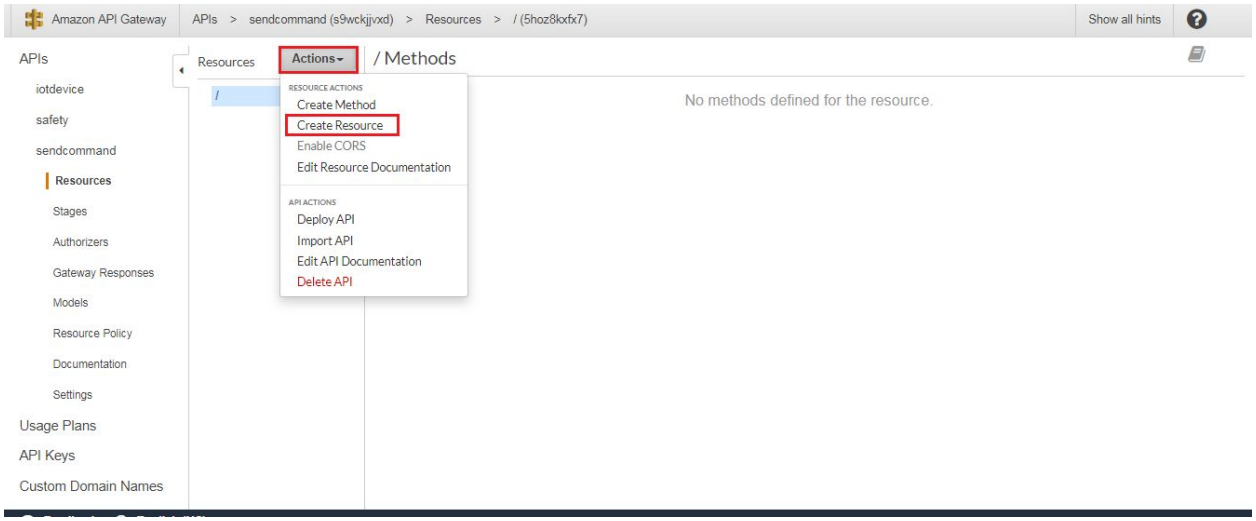

- Double click on that service and you will be redirected to the page where you can create the API. click on create API.




- On the next page select the protocol as rest API, create new API and then give some name to your API and click on create API.

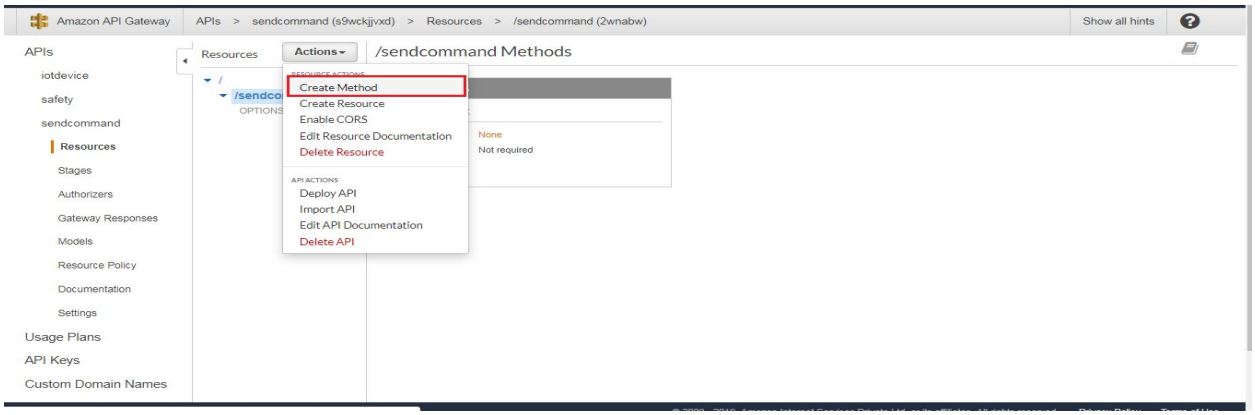



- After creating the API you will get an option of selecting the action. select the action as to create resource.

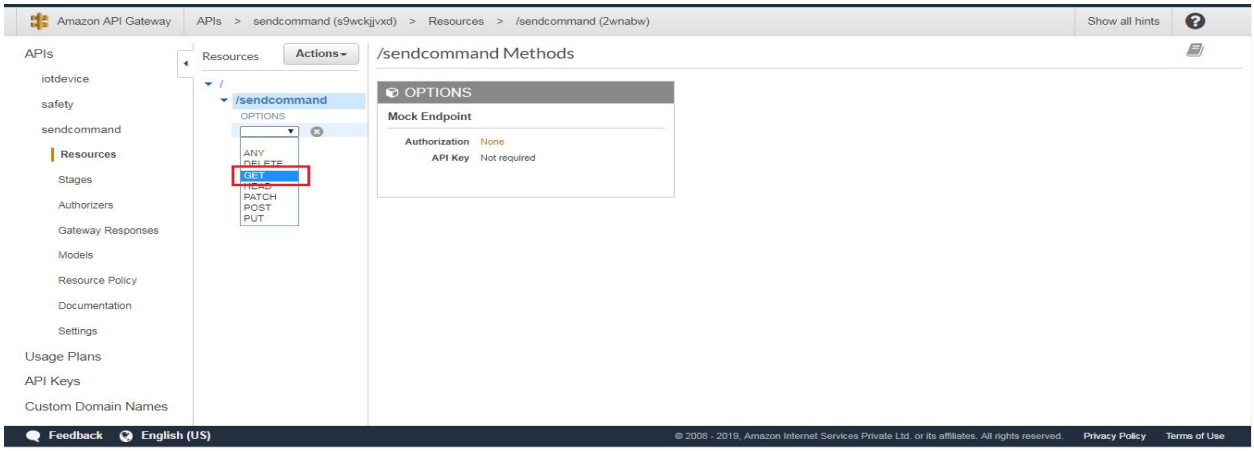



- Give some resource name, enable API Gateway CORS and click on create resource.

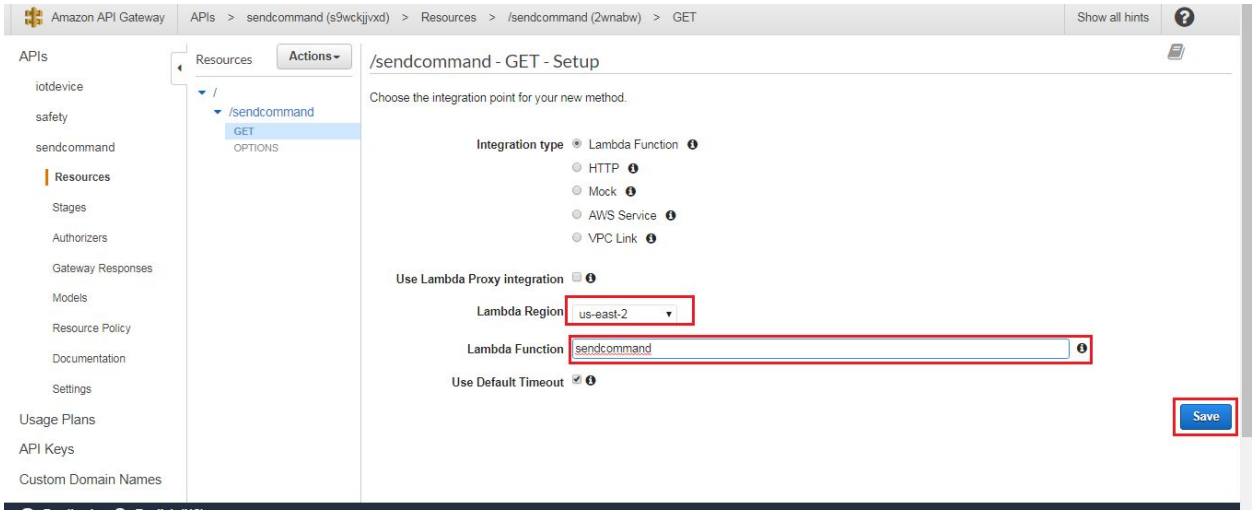

- After creating the resource you need to add the method, go to the actions and click on create method.



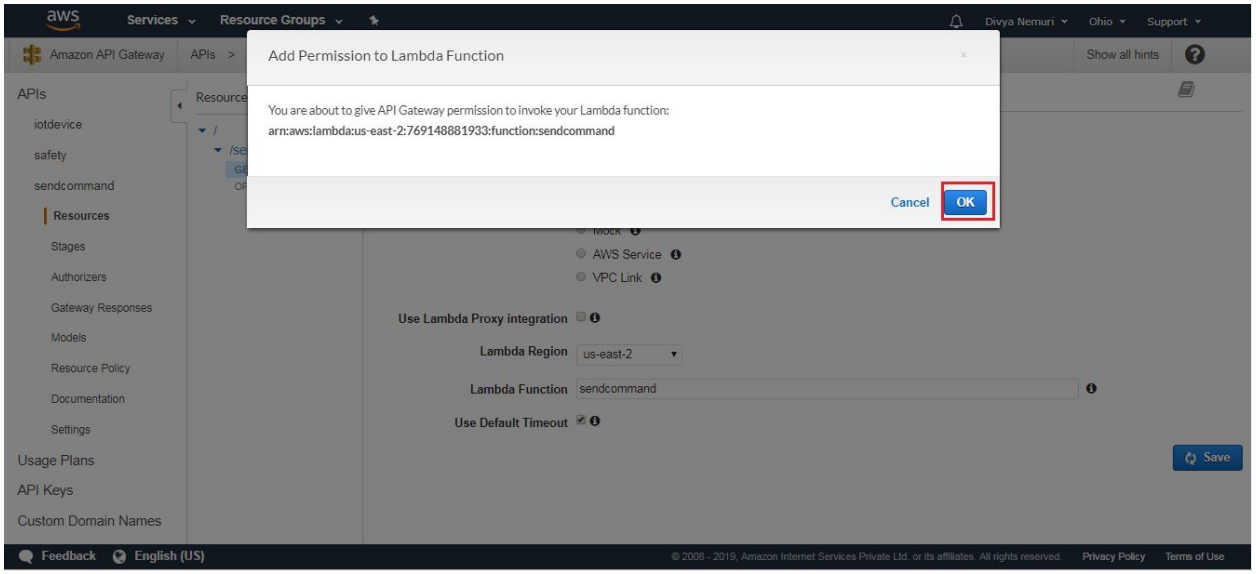

- After creating the method you will get an option of selecting the method.



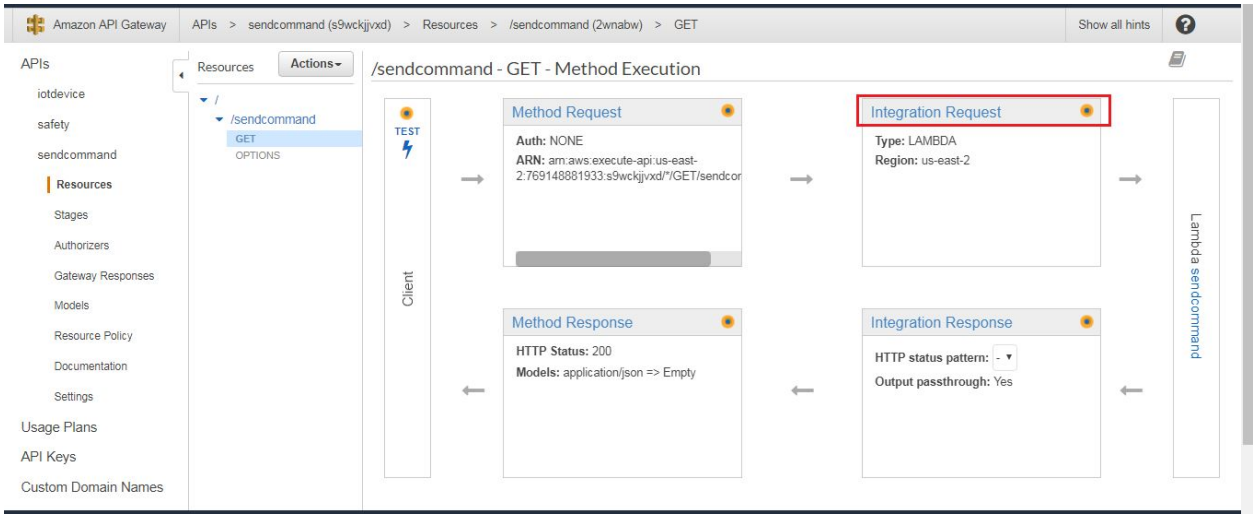
- select GET and click on the tick mark which you got beside.
- Now you need to select the Lambda function region and select the lambda function which you have created and save it.



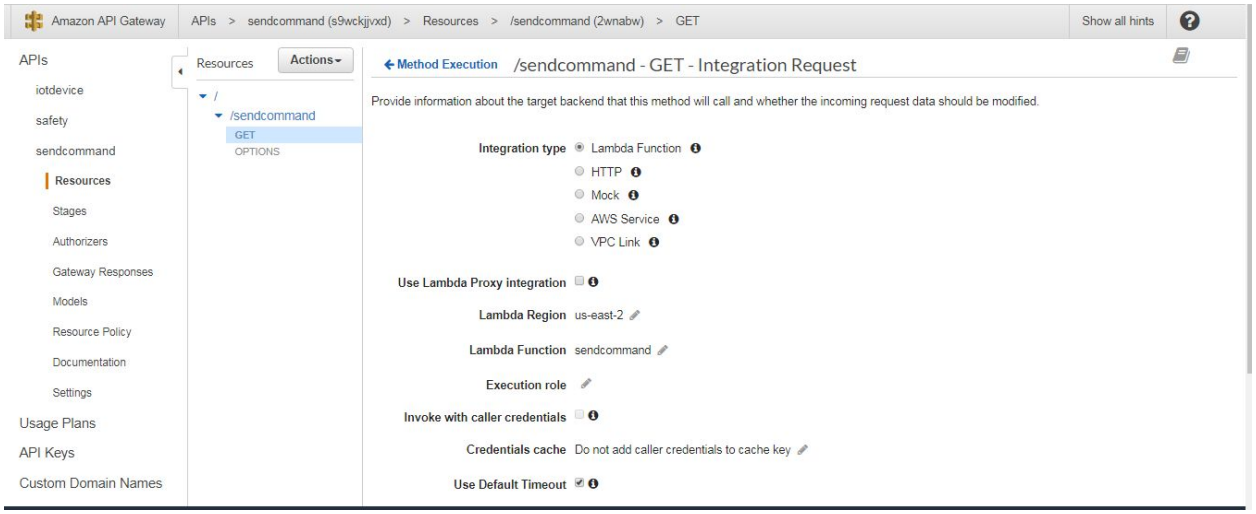

- You will get a popup to add the permission for lambda function, click on ok.

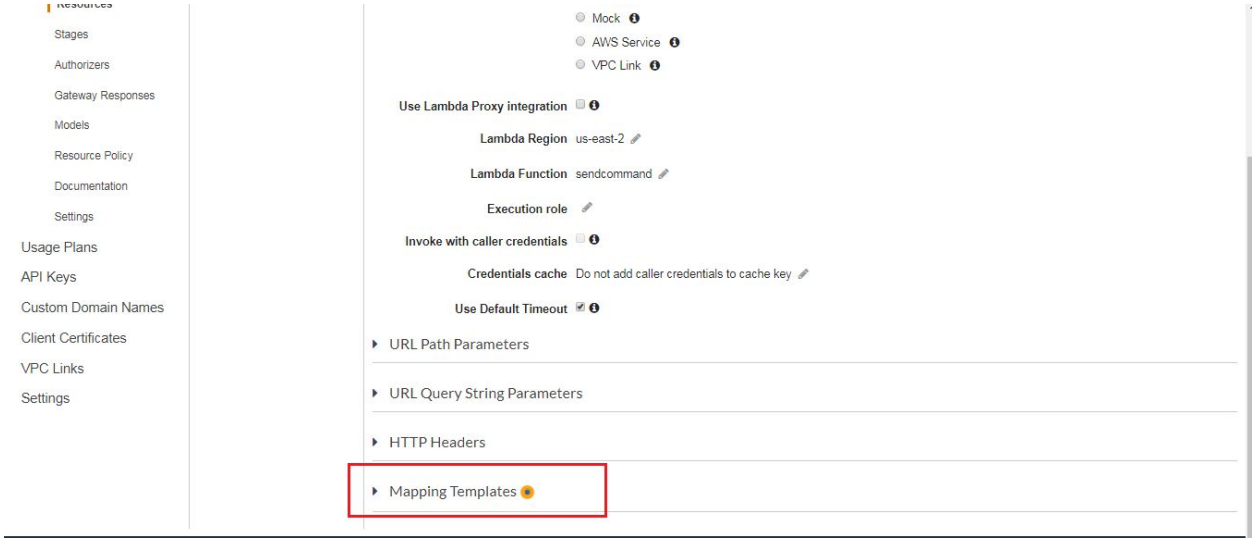



- On this page click on the integration request.

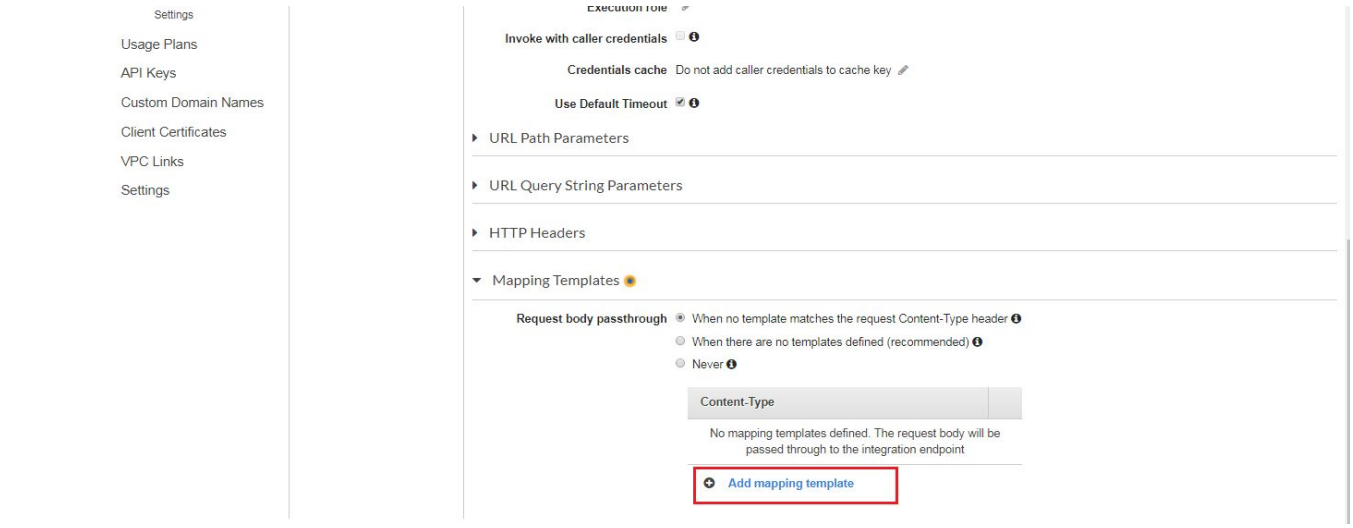



- On that page scroll down to mapping templates and click on that.

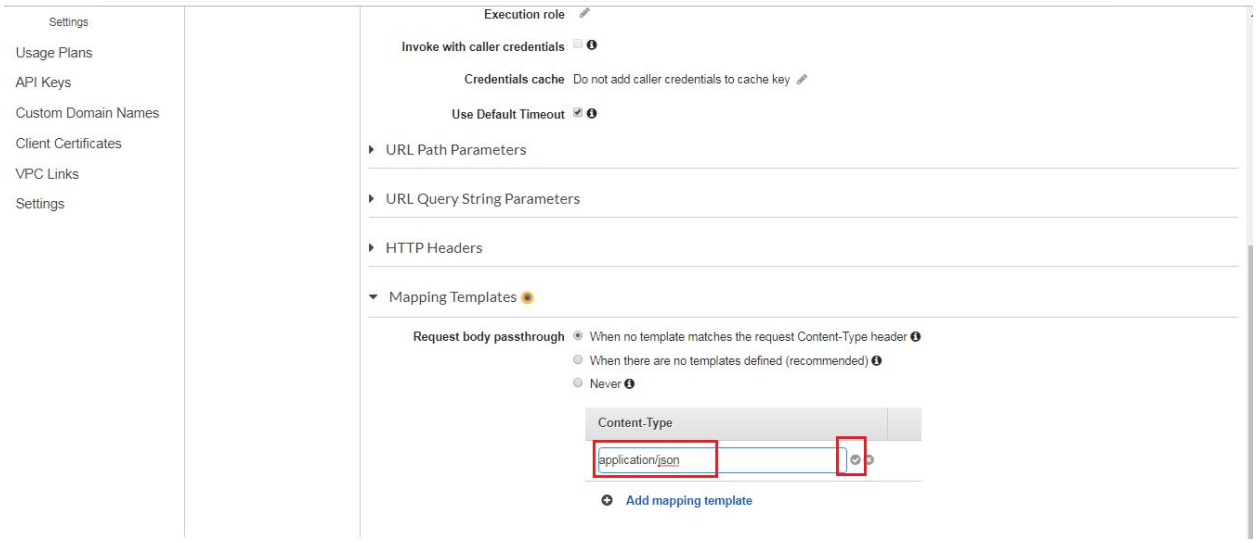


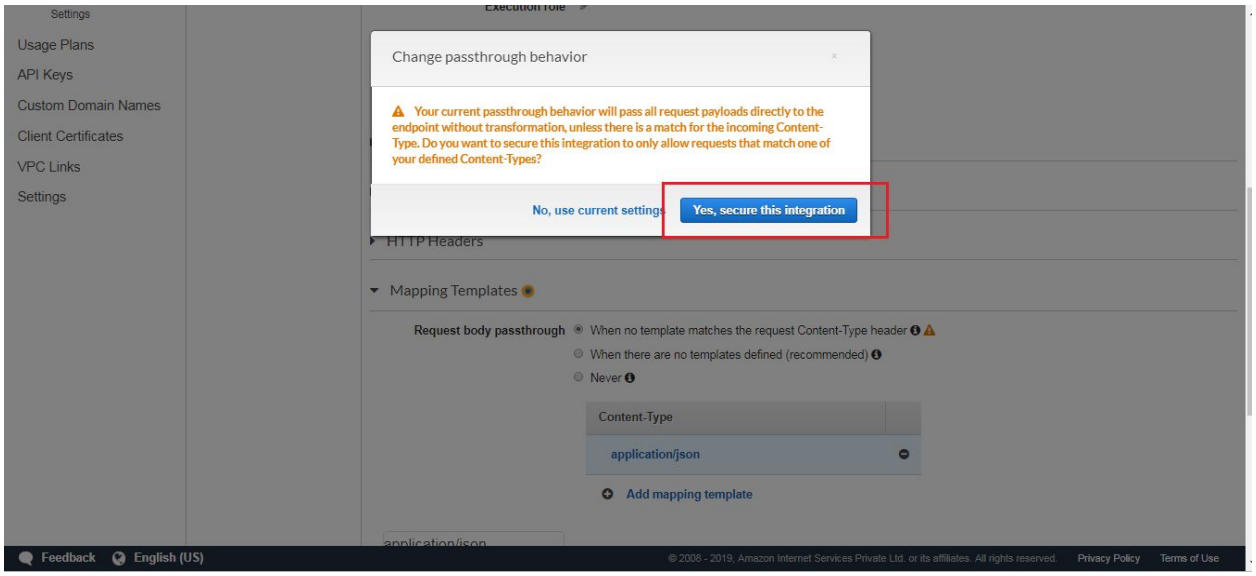
- Click on Add mapping templates




- Add the content type as application/json and click on the tick mark which you got beside.

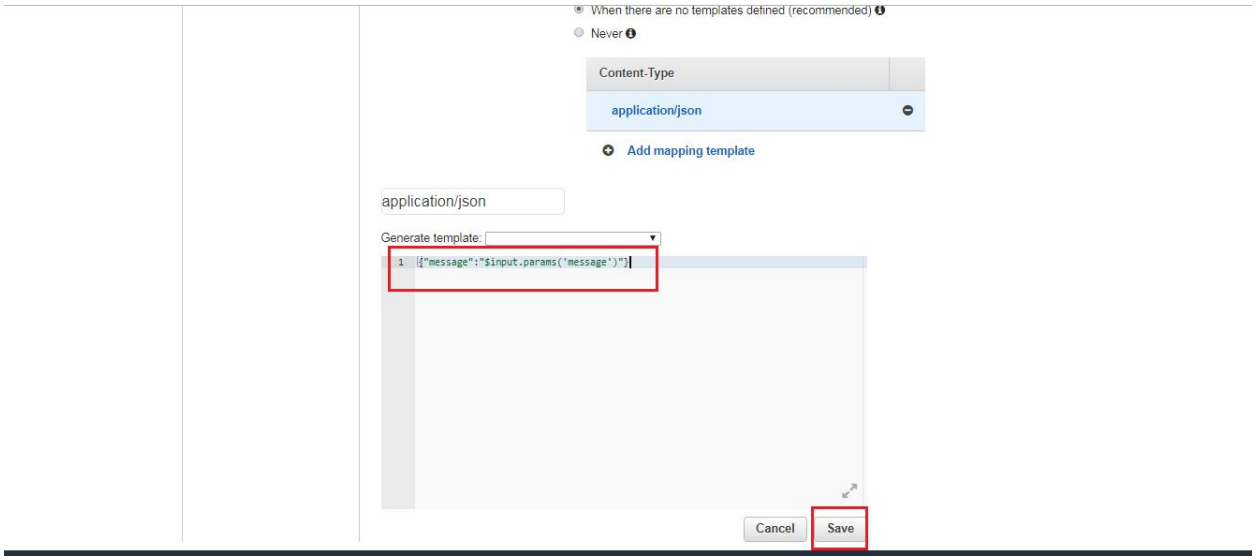



- You will get a popup. Click on yes secure this integration.

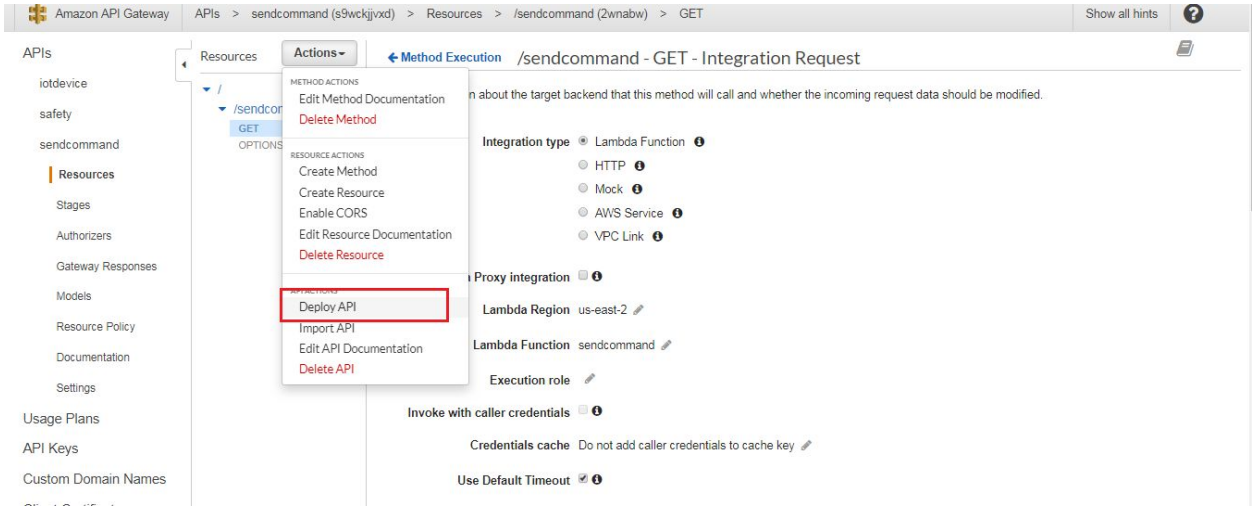



- Now you will get an option of writing the template. There you enter the below code and save it.

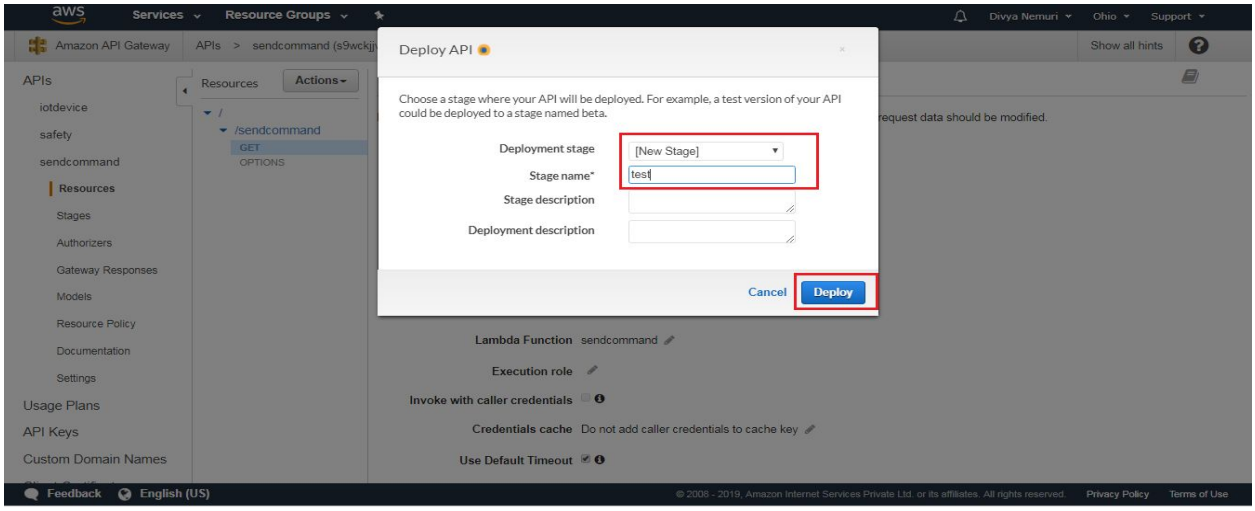
**{“message”:”$input.params(‘message’)”}**



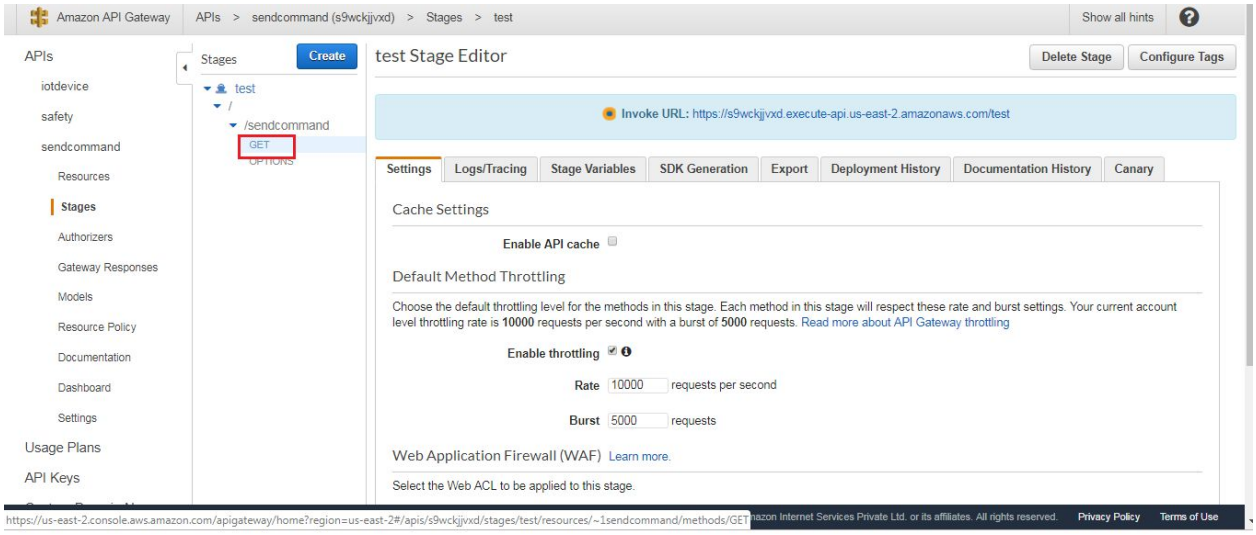

- Now you deploy your API, goto actions and click on deploy API



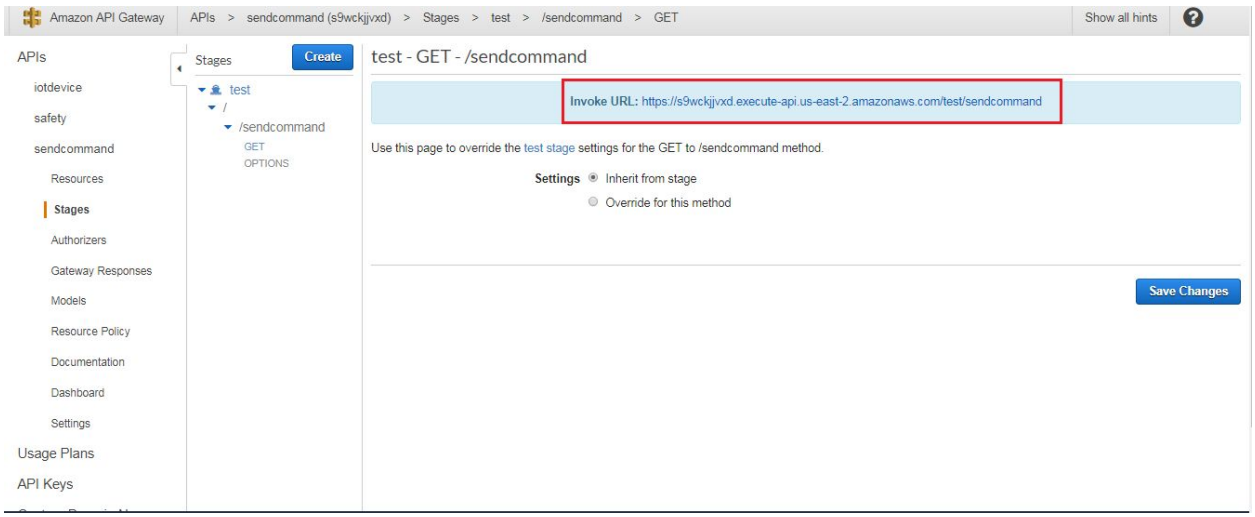

- Select the deployment stage and give some stage name and click on deploy.




- After deploying you will get the stages option in the left side.click on that you will get the options there you click on GET



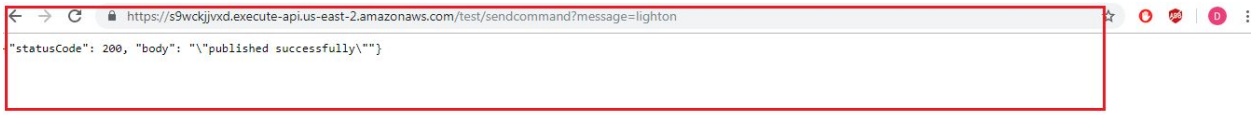

- Now you will get an Invoke URL.
- This is the URL that you can use in your web application or mobile application to send the command to AWS IoT.




- You can test the URL using the browser, open the new tab and paste the URL and give your command

**https://s9wckjjvxd.execute-api.us-east-2.amazonaws.com/test/sendcommand?message=lighton**

- After appending **?message=lighton** to your URL click enter then you need to get a published successfully message.


https://s9wckjjvxd.execute-api.us-east-2.amazonaws.com/test/sendcommand?message=lighton
"statusCode": 200, "body": "\"published successfully\""}