



Invoke Lambda functions with event.json using CLI

1. In this lab we will invoke our lambda function but this time we will use a JSON file to do that.
2. First, we need to create a new folder for this lab again create a commands file that contains all the commands used in this lab.

```
1  Invoke Lambda functions with event.json using CLI
2
3  Multiline char
4  |   for windows = `
5  |   for mac = \
6
7  windows:
8
9  aws lambda invoke \
10 |   --function-name my-function \
11 |   --cli-binary-format raw-in-base64-out \
12 |   --payload file://event.json \
13 |   response.json
```

3. First, we need to create the event.json file VS Code. As you can see below we have created an events file and wrote the code to invoke the lambda function.

```
1  {
2  |   "key": "value"
3  }
```

4. Now we need to run the invoke command and this will invoke our lambda function. But before that, you need to come inside the directory then run the invoke command or else you will get the error.

```
PS C:\Serverless> cd '..\Lab 3 Invoke Lambda functions with event.json using CLI\'
```

5. Below you can see, that our invoke command works properly.

```

` TERMINAL pwsh - Lab 3 Invoke Lambda functions

● PS C:\Serverless> cd '.\Lab 3 Invoke Lambda functions with event.json using CLI\' 
PS C:\Serverless\Lab 3 Invoke Lambda functions with event.json using CLI> aws lambda invoke ` 
>>   --function-name myfirst-function ` 
>>   --cli-binary-format raw-in-base64-out ` 
>>   --payload file://event.json ` 
● >>   response.json 
ExecutedVersion: $LATEST 
StatusCode: 200 

○ PS C:\Serverless\Lab 3 Invoke Lambda functions with event.json using CLI>

```

- Now if I go back to my function and open CloudWatch logs again. So, this time you will see that two logs generated. Open the latest one.

The screenshot shows the AWS CloudWatch Log Stream interface. At the top, there are tabs for 'Log streams', 'Tags', 'Anomaly detection', 'Metric filters', 'Subscription filters', 'Contributor Insights', and 'Data protection'. Below the tabs, it says 'Log streams (2)'. There is a search bar with placeholder text 'Filter log streams or try prefix search' and checkboxes for 'Exact match' and 'Show expired'. A navigation bar shows page 1 of 1. Below the search bar is a table with two rows of log entries. The first row has a checkbox, the text 'Log stream', and a dropdown arrow. The second row has a checkbox, the text '2024/11/11[\$LATEST]c997f146a47a42d7b2db65d1ab51bc2b', and the timestamp '2024-11-11 18:10:36 (UTC+05:30)'. The third row has a checkbox, the text '2024/11/11[\$LATEST]26065a8ef9ca4127a20bae05daf854b1', and the timestamp '2024-11-11 16:59:52 (UTC+05:30)'.

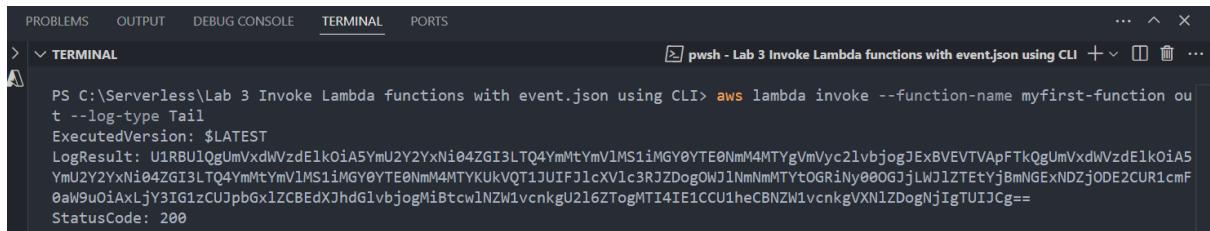
	Log stream	Last event time
<input type="checkbox"/>	2024/11/11[\$LATEST]c997f146a47a42d7b2db65d1ab51bc2b	2024-11-11 18:10:36 (UTC+05:30)
<input type="checkbox"/>	2024/11/11[\$LATEST]26065a8ef9ca4127a20bae05daf854b1	2024-11-11 16:59:52 (UTC+05:30)

- Here you will see that we the logs are generated on behalf of our invocation.

The screenshot shows the AWS CloudWatch Log Events interface. At the top, it shows the path 'CloudWatch > Log groups > /aws/lambda/myfirst-function > 2024/11/11[\$LATEST]c997f146a47a42d7b2db65d1ab51bc2b'. Below the path are buttons for 'Actions', 'Start tailing', and 'Create metric filter'. There is a search bar with placeholder text 'Filter events - press enter to search' and a 'Clear' button. To the right of the search bar are buttons for '1m', '30m', '1h', '12h', 'Custom', 'Local timezone', 'Display', and a gear icon. Below these buttons is a table with columns 'Timestamp' and 'Message'. The table contains several log entries. Some entries have a 'More' link icon to their right. The entries show the Lambda function starting up, receiving a request, processing it, and then stopping.

Timestamp	Message
2024-11-11T18:10:36.571+05:30	INIT_START Runtime Version: nodejs:20.v43 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:d087c1f568d9696a7d95ae9c6...
2024-11-11T18:10:36.714+05:30	START RequestId: dfd2e007-f40e-4950-a108-9acfe8cb7ef5 Version: \$LATEST
2024-11-11T18:10:36.717+05:30	END RequestId: dfd2e007-f40e-4950-a108-9acfe8cb7ef5
2024-11-11T18:10:36.717+05:30	REPORT RequestId: dfd2e007-f40e-4950-a108-9acfe8cb7ef5 Duration: 2.58 ms Billed Duration: 3 ms Memory Size: 128 MB Max Mem...

- Now if we run the command shown below then you will see that this command has given us the log results from the CloudWatch but you will find that this is not in the readable format.

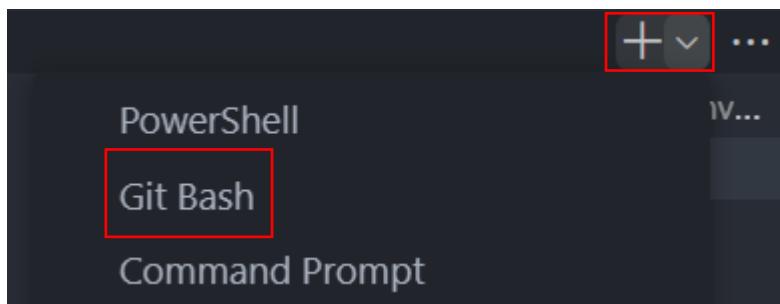


```
PS C:\Serverless\Lab 3 Invoke Lambda functions with event.json using CLI> aws lambda invoke --function-name myfirst-function out --log-type Tail
ExecutedVersion: $LATEST
LogResult: U1RBULQgUmVxdWVzdElkOia5YmU2Y2YxNi04ZG13LTQ4YmMtYmVlMSi1MGY0YTE0NmM4MTYgVmVyc2lvbjogJExBVEVTApFTkQgUmVxdwVzdElkOia5YmU2Y2YxNi04ZG13LTQ4YmMtYmVlMSi1MGY0YTE0NmM4MTYKUkVQT1JUIF3lcXVlc3RJZDogWJ1NmNmMTYtOGRI Ny00OGjJLWJ1ZTETyjBmNGExNDZjODE2CUR1cmF0aW9uAxLjY3IGizCUJpbGx1ZCBEEdXJhdG1vbjogMiBtcwlNZW1vcnkgU216ZTogMTI4IE1CCU1heCBNZW1vcnkgVXN1ZDogNjIgTUIJCg==
StatusCode: 200
```

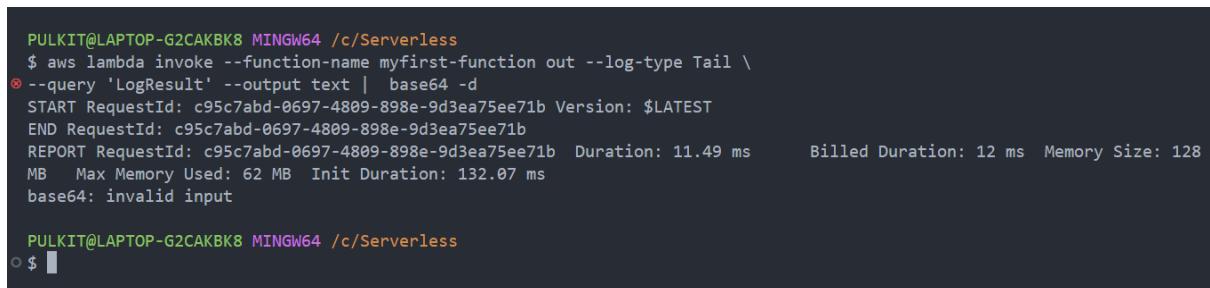
9. So, for that we need to run this command given below. But this thing is we cannot run this command directly on the PowerShell because it is a Linux.
10. So, for this command we need to open the bash terminal inside the VS Code.

```
aws lambda invoke --function-name myfirst-function out --log-type Tail \
--query 'LogResult' --output text | base64 -d
```

11. You need to click on the Plus icon and then choose Git Bash from here.



12. After that run this command to get the latest logs for your function and you will see that the logs are there.



```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 /c/Serverless
$ aws lambda invoke --function-name myfirst-function out --log-type Tail \
--query 'LogResult' --output text | base64 -d
START RequestId: c95c7abd-0697-4809-898e-9d3ea75ee71b Version: $LATEST
END RequestId: c95c7abd-0697-4809-898e-9d3ea75ee71b
REPORT RequestId: c95c7abd-0697-4809-898e-9d3ea75ee71b Duration: 11.49 ms Billed Duration: 12 ms Memory Size: 128 MB Max Memory Used: 62 MB Init Duration: 132.07 ms
base64: invalid input

PULKIT@LAPTOP-G2CAKBK8 MINGW64 /c/Serverless
$
```