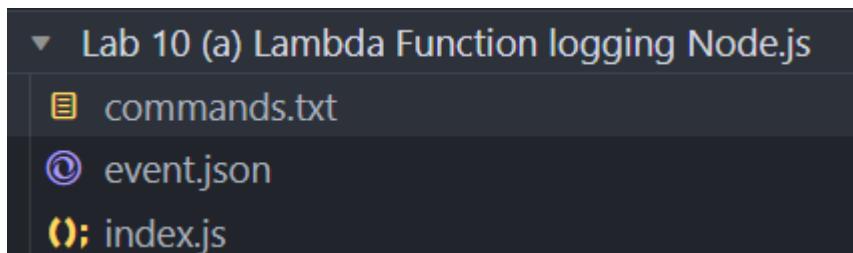




## Lambda Function logging and errors in Node.js

1. In this lab we are going to update our lambda function to have the logging where we will invoke our function and in the CloudWatch logs we will have logs for what we have defined in our index.js code file.
2. First you need to create a lab folder in your VS Code then create the same files which you can see in the snapshot.
3. The event.json file is the same and it is copied from the previous lab.



4. This is the code; we'll be using to update our lambda function. So now create a zip file out of this index.js file.

```
index.js    X
Lab 10 (a) Lambda Function logging Node.js > index.js > handler > handler
1  exports.handler = async (event, context) => {
2
3      console.log("Standart Log")
4      console.info("info Log")
5      console.warn("Warning Log")
6
7      const body =
8          `Function name: ${context.functionName}
9              LogStream name: ${context.logStreamName}
10
11      return body;
12 }
```

5. First, I zipped the index.js file and than ran the command to update my function.

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

> ▾ TERMINAL

PS C:\Serverless\Lab 10 (a) Lambda Function logging Node.js> Compress-Archive index.js function.zip  
 PS C:\Serverless\Lab 10 (a) Lambda Function logging Node.js> aws lambda update-function-code `>> --function-name my-function3`  
 >> --zip-file fileb://function.zip  
 Architectures:  
 - x86\_64  
 CodeSha256: 38+rgktFeN5Br+jxHP+CG8/ofqswyH5EK95rqigIZeE=  
 CodeSize: 283  
 Description: ''  
 EphemeralStorage:  
 Size: 512  
 ● FunctionArn: arn:aws:lambda:us-east-1:878893308172:function:my-function3

6. Now I am going to run the invoke command to have the latest logs. Below you can see that my function has invoked successfully.

```
PS C:\Serverless\Lab 10 (a) Lambda Function logging Node.js> aws lambda invoke `>> --function-name my-function3`  

>> --cli-binary-format raw-in-base64-out `>> --payload file://event.json`  

>> response.json  

>>  

ExecutedVersion: $LATEST  

StatusCode: 200
```

7. In the CloudWatch if I open the latest log events for my function then there are the logs for what we have defined in the code. We have the standard log, Info log, and the warning log.

	Timestamp	Message
No older events at this moment. <a href="#">Retry</a>		
▶	2024-11-12T18:19:44.389+05:30	INIT_START Runtime Version: nodejs:18.v49 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:
▶	2024-11-12T18:19:44.561+05:30	START RequestId: 0a3cc5e1-d65e-4504-b585-ef82901ac65a Version: \$LATEST
▼	2024-11-12T18:19:44.562+05:30	2024-11-12T12:49:44.562Z 0a3cc5e1-d65e-4504-b585-ef82901ac65a INFO Standard Log
	2024-11-12T12:49:44.562Z	0a3cc5e1-d65e-4504-b585-ef82901ac65a INFO Standard Log
▼	2024-11-12T18:19:44.563+05:30	2024-11-12T12:49:44.563Z 0a3cc5e1-d65e-4504-b585-ef82901ac65a INFO info Log
	2024-11-12T12:49:44.563Z	0a3cc5e1-d65e-4504-b585-ef82901ac65a INFO info Log
▼	2024-11-12T18:19:44.563+05:30	2024-11-12T12:49:44.563Z 0a3cc5e1-d65e-4504-b585-ef82901ac65a WARN Warning Log
	2024-11-12T12:49:44.563Z	0a3cc5e1-d65e-4504-b585-ef82901ac65a WARN Warning Log
▶	2024-11-12T18:19:44.578+05:30	END RequestId: 0a3cc5e1-d65e-4504-b585-ef82901ac65a

## 😢 Errors in Node.js

8. Now we will again update our function to have an error in it. First, we created the lab folder in VS Code then we created the commands text file. The event.json file was copied from the previous lab.

## ▼ Lab 10 (b) Lambda Function errors in Node.js

- ❑ commands.txt
- ❷ event.json
- ❸ index.js

9. Below is the code used in index.js.

```
❸ index.js X  
Lab 10 (b) Lambda Function errors in Node.js > ❸ index.js > ❶ handler > ❶ handler  
1  ✓ exports.handler = async (event, context) => {  
2      ...  
3      throw Exception('error from lambda code')  
4  }
```

10. Now the steps are the same as above first we will zip the index.js file and then update our lambda function then we will invoke it as well.

```
PS C:\Serverless\Lab 10 (a) Lambda Function logging Node.js> cd ..  
● PS C:\Serverless> cd '.\Lab 10 (b) Lambda Function errors in Node.js\'  
PS C:\Serverless\Lab 10 (b) Lambda Function errors in Node.js> Compress-Archive index.js function.zip  
● >  
○ PS C:\Serverless\Lab 10 (b) Lambda Function errors in Node.js> aws lambda update-function-code `  
    --function-name my-function3`  
    --zip-file fileb://function.zip  
    --  
Architectures:  
- x86_64  
CodeSha256: DcGhCFYEBT8l00NKHcrX0T9JF4BWmeiE8iy3l4U5o8c=  
CodeSize: 202  
Description: ''
```

11. Here you can see that when we invoked our lambda function we get the status code 200 but we also got the function error unhandled.

```
PS C:\Serverless\Lab 10 (b) Lambda Function errors in Node.js> aws lambda invoke `  
    --function-name my-function3`  
    --cli-binary-format raw-in-base64-out`  
    --payload file://event.json`  
    --  
    response.json  
    --  
ExecutedVersion: $LATEST  
FunctionError: Unhandled  
StatusCode: 200  
  
PS C:\Serverless\Lab 10 (b) Lambda Function errors in Node.js> █
```

12. In the response file as well, you can read the error.

```
commands.txt      response.json X
Lab 10 (b) Lambda Function errors in Node.js > response.json > ...
1  {"errorType": "ReferenceError", "errorMessage": "Exception is not defined", "trace": ["ReferenceError: Exception is not defined", "    at exports.handler (/var/task/index.js:2:5)", "    at Runtime.handleOnceNonStreaming (file:///var/runtime/index.mjs:1173:29)"]}
```

13. If you open the CloudWatch logs then you can see the error.

The screenshot shows the AWS CloudWatch Log Events interface. At the top, there's a filter bar with a search input, time range buttons (Clear, 1m, 30m, 1h, 12h, Custom), a Local timezone dropdown, and a Display dropdown. Below the filter bar is a table header with columns: ▶, Timestamp, and Message. The table body contains several log entries. The first entry is collapsed, showing only the timestamp and message summary. The second entry is collapsed, showing the runtime version and request ID. The third entry is expanded, showing the full log message with detailed error information. The expanded message includes the error type (ReferenceError), message (Exception is not defined), and a stack trace pointing to the exports.handler function in index.js at line 2, character 5. The log entry ends with an invoke error status. The fourth entry is collapsed, showing the end request ID. The fifth entry is collapsed, showing the report request ID with performance metrics.

▶	Timestamp	Message
▶	2024-11-12T18:26:31.475+05:30	INIT START Runtime Version: nodejs:18.v49 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:13821268cdb8b1fd3647b6b7f04...
▶	2024-11-12T18:26:31.651+05:30	START RequestId: e60a36c1-d6cc-438b-b6e0-91155736dd1c Version: \$LATEST
▼	2024-11-12T18:26:31.653+05:30	2024-11-12T12:56:31.653Z e60a36c1-d6cc-438b-b6e0-91155736dd1c ERROR Invoke Error {"errorType": "ReferenceError", "errorMessage": "Exception is not defined", "stack": ["ReferenceError: Exception is not defined", "    at exports.handler (/var/task/index.js:2:5)", "    at Runtime.handleOnceNonStreaming (file:///var/runtime/index.mjs:1173:29)"]}
▶	2024-11-12T12:56:31.653Z	e60a36c1-d6cc-438b-b6e0-91155736dd1c <b>ERROR</b> Invoke <b>Error</b>
▶	2024-11-12T18:26:31.707+05:30	END RequestId: e60a36c1-d6cc-438b-b6e0-91155736dd1c
▶	2024-11-12T18:26:31.707+05:30	REPORT RequestId: e60a36c1-d6cc-438b-b6e0-91155736dd1c Duration: 55.84 ms Billed Duration: 56 ms Memory Size: 128 MB Max Mem...