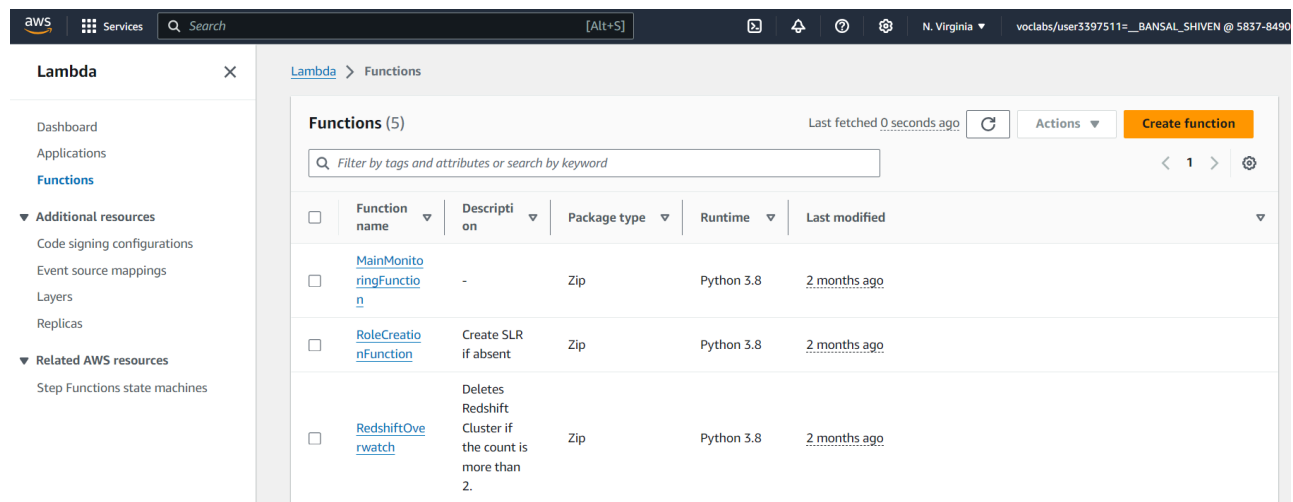


Experiment 11

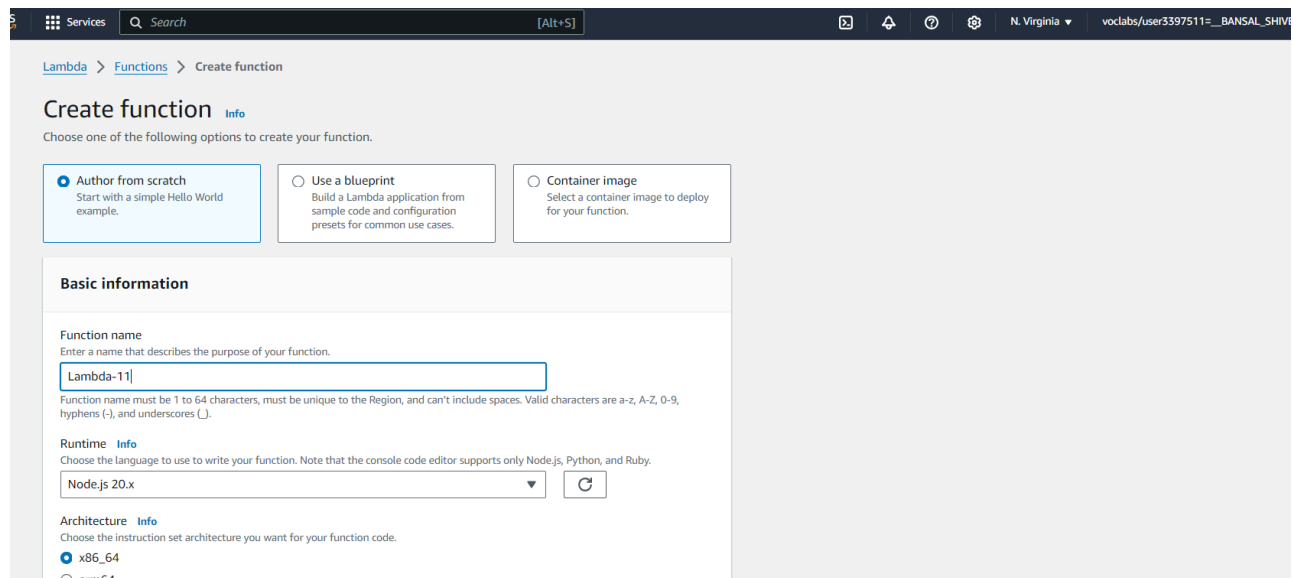
Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

Steps:

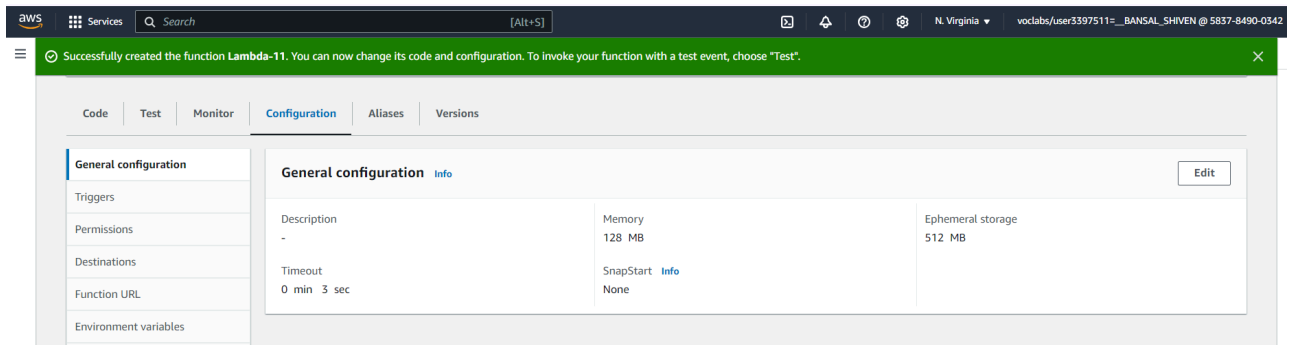
Step 1: On your AWS console, click on 'Lambda' in the services section and click on 'Create function'.



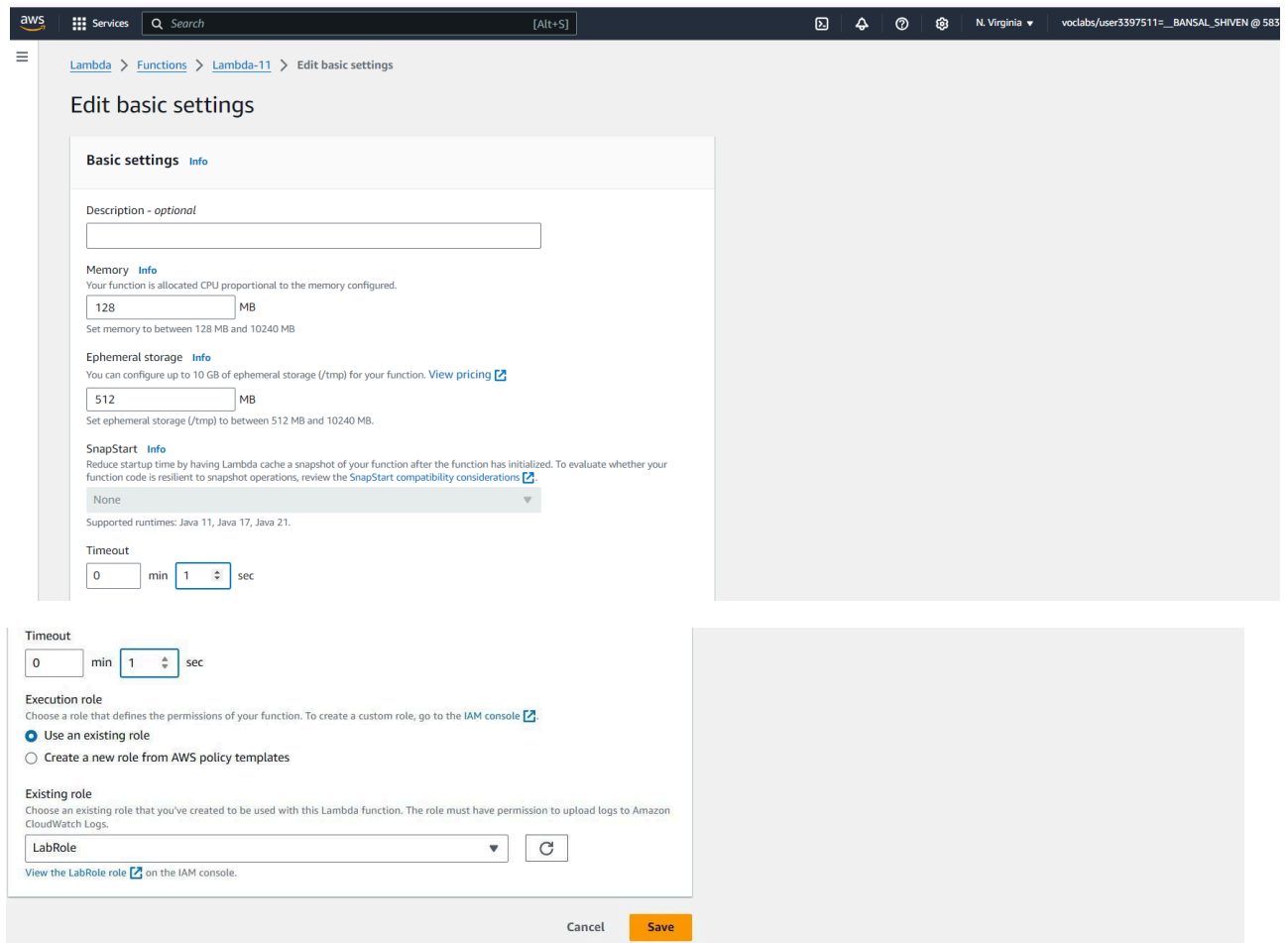
Step 2: Give your Lambda function a name. Select the language to use to write your function (Node.js is the default and what we will use in this experiment). Keep other options as default.



Step 3: The general configuration of the function is visible in the 'Configuration' tab. To change the configuration, click on 'Edit'.

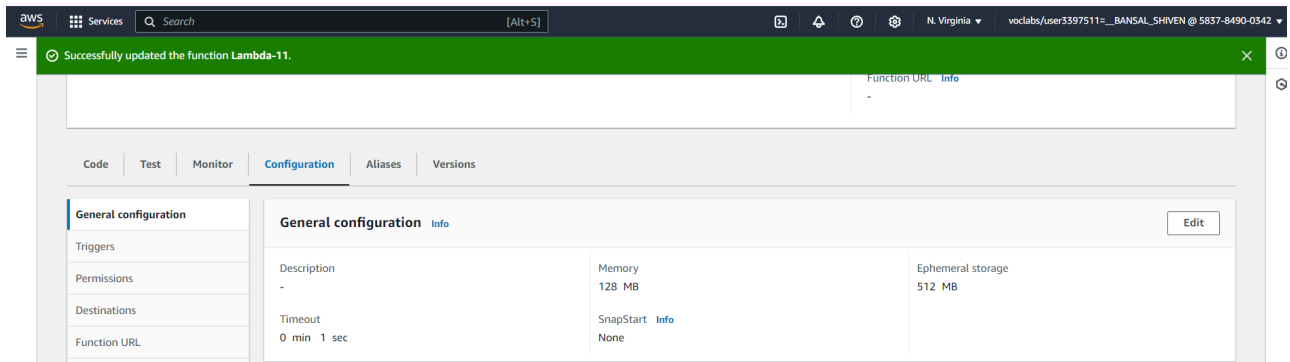


You can change the various parameters of the configuration as per your needs. Here, we can change the 'Timeout' period to 1 second as it's sufficient for our function for now. 'Timeout' is the time for which a function can be running before it gets forcibly terminated.

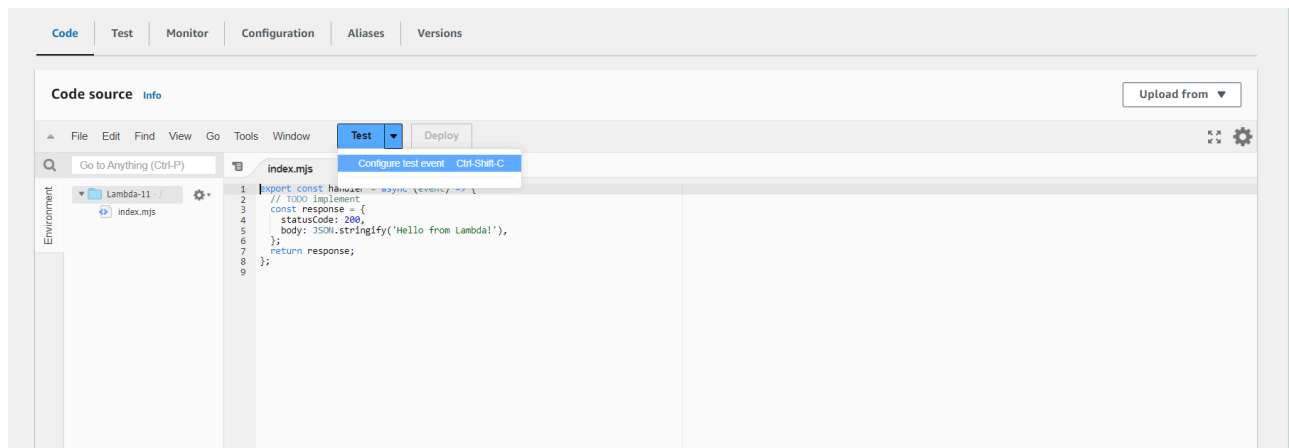


After making the required changes, click on 'Save'.

The changes in the general configuration are visible in the function.



Step 4: In the 'Code source' section, click on the arrow next to the 'Test' button and click on 'Configure test event'.



Step 5: Give your test event a name, keep all other options as default and click on 'Save'.

Configure test event

A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result.

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action

Create new event

Edit saved event

Event name

LambdaEvent-11

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

Private

This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

Shareable

This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional

hello-world

Event JSON

Format JSON

```

1 {
2   "key1": "value1",
3   "key2": "value2",
4   "key3": "value3"
5 }

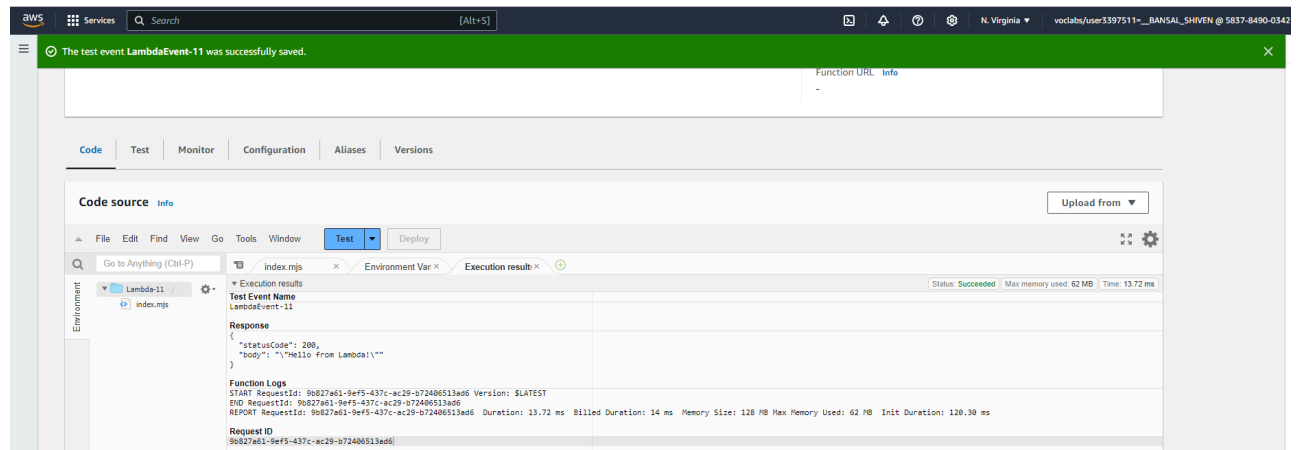
```

Cancel

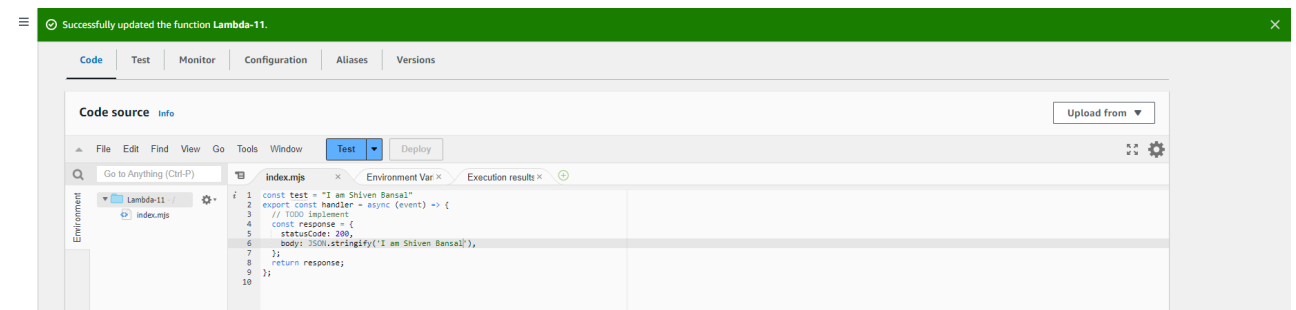
Invoke

Save

Step 6: Click on the 'Test' button. The following output appears.



Step 7: You can make changes in the code to observe the difference in the output. Here, we change the code to display a different string as such:-



Once the changes are made, click on 'Deploy'.

Step 8: Click on 'Test' and observe how the output after the changes differs from the output before the changes.

Successfully updated the function Lambda-11.

Code source info Upload from ▼

File Edit Find View Go Tools Window Test ▼ Deploy

Go to Anything (Ctrl-P)

Environment: Lambda-11 index.mjs

Execution results: Status: Succeeded Max memory used: 62 MB Time: 5.87 m

Test Event Name: LambdaEvent-11

Response:

```
{
  "statusCode": 200,
  "body": "\nI am Shiven Bansal\n"
}
```

Function Logs:

```
START RequestId: 204b9a7c-9093-4101-9dd9-e18f374f2266 Version: $LATEST
END RequestId: 204b9a7c-9093-4101-9dd9-e18f374f2266
REPORT RequestId: 204b9a7c-9093-4101-9dd9-e18f374f2266 Duration: 5.87 ms Billed Duration: 6 ms Memory Size: 128 MB Max Memory Used: 62 MB Init Duration: 141.93 ms
```

Request ID: 204b9a7c-9093-4101-9dd9-e18f374f2266

Conclusion:

In this experiment, I explored AWS Lambda and created my first Lambda function using Node.js. I learned how to configure a Lambda function, including selecting execution roles and adjusting parameters like the timeout period. I also tested the function by creating and executing a test event. After making changes to the code, I deployed the new version and observed how the output changed. This experiment gave me hands-on experience with Lambda's workflow, enhancing my understanding of serverless computing and how to deploy and test functions on AWS.