Amazon Web Services

Lambda

ankitnarula1991@gmail.com

Lambda

- AWS Lambda is a service which computes the code without any server. It is said to be server less compute. The code is executed based on the response of events in AWS services.

- To get working with AWS Lambda, we just have to push the code in AWS Lambda service. All other tasks and resources such as infrastructure, operating system, maintenance of server, code monitoring, logs and security is taken care by AWS.

Lambda - functions supports code:

- Node.js
- Python
- Ruby
- Java
- Go
- C#
- Powershell

Lambda - Advantages

- **Ease of Working with Code**: AWS Lambda gives you the infrastructure to upload your code. It takes care of maintaining the code and triggers the code whenever the required event happens. It allows you to choose the memory and the timeout required for the code. AWS Lambda can also execute parallel requests as per the event triggers.

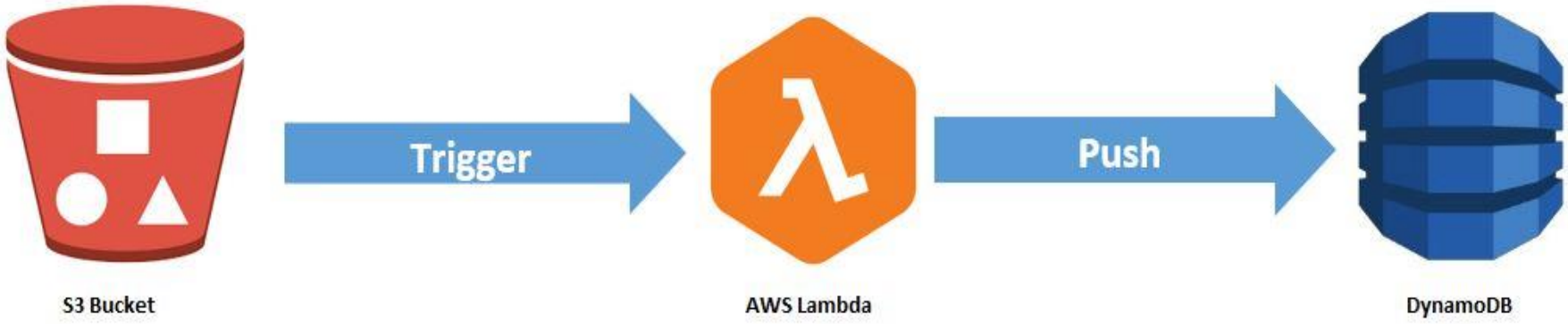
- **Logs Provision**: AWS Lambda gives the details of number of times a code was executed and time taken for execution, the memory consumed etc. AWS CloudWatch collects all the logs, which helps in understanding the execution flow and in the debugging of the code.

- **Billing based on Usage**: AWS Lambda billing is done on memory usage, request made and the execution, which is billed in increments of minimum 100ms. So for a 500ms execution, the billing will be after every 100ms. If you specify your AWS lambda code to be executed in 500ms and the time taken to execute is just 200ms, AWS will bill you only for the time taken that is 200ms of execution instead of 500ms. AWS always charges for the execution time used. You need not pay if the function is not executed.

- **Multi Language Support:** AWS Lambda supports popular languages such as Node. js, Python, Java, C# and Go. These are widely used languages and any developer will find it easy to write code for AWS Lambda.

Lambda - Integration with other AWS Services

- AWS API Gateway
- Amazon S3
- AWS Cloud Formation
- AWS Kinesis
- AWS CloudWatch Events
- AWS DynamoDB
- AWS SNS
- **Lambda Quota: Click Here**



Lambda

- **Create IAM Role**
- Go to IAM Service
- Click on Roles
- Click on Create Role
- Select AWS Services
- Select Lambda & Click on Next: Permissions
- Search AmazonDynamoDBFullAccess

Lambda

- Enter the Role Name
- Click on Create Role
- **Create S3 Bucket**
- Unblock all public access and acknowledge the settings
- **Create DynamoDB Table**
- Enter the table Name (newtable)
- Enter partition key (unique)

Lambda

- Click on Create table
- Go to Lambda Service
- Click on Create Function
- Enter the Function Name
- Select Runtime (Python 3.6)
- Click on Change default execution role
- Select the use an existing role
- Select the role we have created

Lambda

- Click on Create function
- Lambda Python Code: **Click Here**
- Click on Deploy
- Click on Add Trigger
- Select trigger as S3
- Select the bucket
- Select event type as per requirement (All Object Create Events)

Lambda

- Check the acknowledge
- Click on Add
- Upload the Object in S3 Bucket & Give Public Access to Object
- Now check the DynamoDB table
- Click on Run

ankitnarula1991@gmail.com