AWS LAMBDA (Serverless Computing)

What is Lambda ?

- It is a compute service that allows you to run application code/logic without provision a server

Why to use Lambda ?

- It doesn't require any server to be provisioned , no maintenance for underlying infrastructure
- You can run your application code/logic without wasting time on provisioning server or needed infrastructure
- You do not have to think about autoscaling and high availability, everything will be managed by AWS
- Without any additional overhead you will always get consistent performance just by selecting right amount of memory size to run your function
- Well integrated with various other AWS services which enables you to automate many automation or build serverless application with less effort
- Only pay for the used computing time and it charged for every 100ms you executes your code and the number of times it executed
- Zero administration

AWS Lambda Features :

- You can create a backend service for your application using Lambda
- It has bult-in fault tolerance
- Using EFS for Lambda you can read, write and persist large volume of data at low latency and any scale
- Using Lambda@Edge you can run codes across regions globally in response to CloudFront events
- Integrated with AWS step function which enable you to build a stateful and long running processes for your application
- You just need to required amount of memory to allocate and lambda will allocate the proportional CPU, network bandwidth and storage automatically
- You just pay for the execution duration rather than underlying infrastructure or the server unit
- It also supported with Compute savings plans and EC2 instance savings plans for further cost savings
- Lambda supported with VPC which means you can run Lambda functions within your private network

Supported Languages :

- Java
- Go
- PowerShell
- Node.js
- C#,
- Python
- Ruby

How Lambda Works ?

- First you upload your code/ function
- Then lambda will execute the code on your behalf
- Once code is invoked lambda automatically start provisioning and managing the required servers in the backend

When to use LAMBDA and when to use EC2 ?

- EC2 Instance:
  - IaaS based service model
  - You can run any languages
  - Pricing model is per second
  - For a complex and multipurpose application
  - When you need high processing services functions
  - When you need more computing power
  - Required for long running processes
  - When you need a customized AMI or OS
- Lambdas Function :
  - PaaS based service model
  - Supported limited languages
  - Pricing model is per milli second
  - For modular application with shorter computing time
  - When you need to run function as a service
  - For simple and light weight applications, logics and automation
  - For log analysis

Important Terms of Lambda :

- Function :
  - Using function you can invoke your code to run on lambda
  - Using function you can upload or write your code
- Runtime :
  - It allows functions in different languages to run the code and it sits between Lambda service and function/code
- Event :
  - Its JSON format document which contains data for the function to process
- Event Source/Trigger :
  - AWS services such as SNS, CloudWatch that trigger your function to execute the logic
  - you can see then on left side of Lambda function
- Downstream Resource :
  - AWS service such as Dynamo, S3 that lambda function can calls once the function is triggered
  - You can see then on right side of the Lambda function
- Concurrency :
  - Number of request that your function is serving at a given time

- At a time you can run 1000 times simultaneously and it's the maximum limit

Some Examples Triggers :

- Any changes or events occurred on the S3 bucket and DynamoDB
- Run codes in response to HTTP(S) requests using API gateway

Lambda Function Configuration

- Consists of code and associated dependencies
- You can change configuration once created using API calls
- You only specify required memory information and other required resources like CPU and Network bandwidth will be allocated automatically based on the allocated memory
- You can update the configuration and request additional memory in 64MB increments from 128MB (minimum) to 3008MB (maximum)
- Functions larger than 1536MB are allocated multiple CPU threads
- Default execution time is 3 second and maximum is 900 seconds (15 minutes)
- To prevent running function indefinitely you need to specify a timeout which prevent additional cost, once it breaches the timeout it terminate the function
- IAM role has to be created which assumes when it execute function on your behalf to access aws services