ASSIGNMENT : 8

**About API GATEWAY:**

Amazon API Gateway is an AWS service for creating, publishing, maintaining, monitoring, and securing REST, HTTP, and WebSocket APIs at any scale. API developers can create APIs that access AWS or other web services, as well as data stored in the [AWS Cloud](https://aws.amazon.com/what-is-cloud-computing/). As an API Gateway API developer, you can create APIs for use in your own client applications. Or you can make your APIs available to third-party app developers.

API Gateway creates RESTful APIs that:

* Are HTTP-based.
* Enable stateless client-server communication.
* Implement standard HTTP methods such as GET, POST, PUT, PATCH, and DELETE.
* **Architecture of API Gateway**
* The following diagram shows API Gateway architecture.
* 
                  API Gateway architecture diagram
              
* This diagram illustrates how the APIs you build in Amazon API Gateway provide you or your developer customers with an integrated and consistent developer experience for building AWS serverless applications. API Gateway handles all the tasks involved in accepting and processing up to hundreds of thousands of concurrent API calls. These tasks include traffic management, authorization and access control, monitoring, and API version management.
* API Gateway acts as a "front door" for applications to access data, business logic, or functionality from your backend services, such as workloads running on Amazon Elastic Compute Cloud (Amazon EC2), code running on AWS Lambda, any web application, or real-time communication applications.

LAMDA:

AWS Lambda is a compute service that lets you run code without provisioning or managing servers.

Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, and logging. With Lambda, all you need to do is supply your code in one of the language runtimes that Lambda supports.

You organize your code into Lambda functions. The Lambda service runs your function only when needed and scales automatically. You only pay for the compute time that you consume—there is no charge when your code is not running.

Lambda is an ideal compute service for application scenarios that need to scale up rapidly, and scale down to zero when not in demand. For example, you can use Lambda for:

* **File processing:** Use Amazon Simple Storage Service (Amazon S3) to trigger Lambda data processing in real time after an upload.
* **Stream processing:** Use Lambda and Amazon Kinesis to process real-time streaming data for application activity tracking, transaction order processing, clickstream analysis, data cleansing, log filtering, indexing, social media analysis, Internet of Things (IoT) device data telemetry, and metering.
* **Web applications:** Combine Lambda with other AWS services to build powerful web applications that automatically scale up and down and run in a highly available configuration across multiple data centers.
* **IoT backends:** Build serverless backends using Lambda to handle web, mobile, IoT, and third-party API requests.
* **Mobile backends:** Build backends using Lambda and Amazon API Gateway to authenticate and process API requests. Use AWS Amplify to easily integrate with your iOS, Android, Web, and React Native frontends.

When using Lambda, you are responsible only for your code. Lambda manages the compute fleet that offers a balance of memory, CPU, network, and other resources to run your code. Because Lambda manages these resources, you cannot log in to compute instances or customize the operating system on provided runtimes.

DEPLOYING HELLOWORLD API ON AWS LAMBDA:

Import json

Def lambda\_handler (event,context):

Print (“testing python in lambda”)

Response = {

“statuscode” : 200;

‘body’ : json.dumps (‘hello world from lambda!’)

}

Return response