St. Francis Institute of Technology, Mumbai-400 103
Department Of Information Technology

A.Y. 2021-2022
Class: TE-ITA/B, Semester: V

Subject: **Advanced DevOps Lab**

Experiment – 11: To understand AWS Lambda, its workflow, and to create the first Lambda function using Python/Java.

1. **Aim:** To write the first Lambda function using Python/Java/Node.js.
2. **Objectives:** Aim of this experiment is that, the students will learn:
   - Serverless cloud concept and how to create Lambda function in various languages
   - Invoke Lambda function
   - Monitoring AWS Lambda
3. **Outcomes:** After study of this experiment, the students will learn following:
   - Introduction to Serverless computing
   - Creating Lambda function
   - Testing Lambda function
   - Monitoring and Logging Lambda function
4. **Prerequisite:** Knowledge of Python/Java/Node.js and AWS console.
5. **Requirements:** AWS account, browser, Personal Computer, Windows operating system, Internet Connection, Google Doc.
6. **Pre-Experiment Exercise:**
   Brief Theory: Refer shared material
7. **Laboratory Exercise**
   A. Procedure:
   a. Answer the following:
   - Explain Serverless concept?

   → Serverless computing is a method of providing backend services on an as-used basis. A serverless provider allows users to write and deploy code without the hassle of worrying about the underlying infrastructure. A company that gets backend services from a serverless vendor is charged based on their computation and do not have to reserve and pay for a fixed amount of bandwidth or number of servers, as the service is auto-scaling. Note that despite the name serverless, physical servers are still used but developers do not need to be aware of them.

   Serverless computing allows developers to purchase backend services on a flexible 'pay-as-you-go' basis, meaning that developers only have to pay for the services they use. This is like switching from a cell phone data plan with a monthly fixed limit, to one that only charges for each byte of data that actually gets used.

   The term 'serverless' is somewhat misleading, as there are still servers providing these backend services, but all of the server space and infrastructure concerns are handled by the vendor. Serverless means that the developers can do their work without having to worry about servers at all.

Advantages are:

1. Lower costs - Serverless computing is generally very cost-effective, as traditional cloud providers of backend services (server allocation) often result in the user paying for unused space or idle CPU time.
2. Simplified scalability - Developers using serverless architecture don't have to worry about policies to scale up their code. The serverless vendor handles all of the scaling on demand.
3. Simplified backend code - With FaaS, developers can create simple functions that independently perform a single purpose, like making an API call.
4. Quicker turnaround - Serverless architecture can significantly cut time to market. Instead of needing a complicated deploy process to roll out bug fixes and new features, developers can add and modify code on a piecemeal basis.

● What are the applications of AWS Lambda?
→ An AWS Lambda application is a combination of Lambda functions, event sources, and other resources that work together to perform tasks. You can use AWS CloudFormation and other tools to collect your application's components into a single package that can be deployed and managed as one resource. Applications make your Lambda projects portable and enable you to integrate with additional developer tools, such as AWS CodePipeline, AWS CodeBuild, and the AWS Serverless Application Model command line interface (SAM CLI).

Use cases:

1. Data processing

You can use AWS Lambda to execute code in response to triggers such as changes in data, shifts in system state, or actions by users. Lambda can be directly triggered by AWS services such as S3, DynamoDB, Kinesis, SNS, and CloudWatch. It can also connect to existing EFS file systems or be orchestrated into workflows by AWS Step Functions. This allows you to build a variety of real-time serverless data processing systems.

2. Real-time file processing

You can use Amazon S3 to trigger AWS Lambda to process data immediately after an upload. You can also connect to an existing Amazon EFS file system directly, which enables massively parallel shared access for large scale file processing. For example, you can use Lambda to thumbnail images, transcode videos, index files, process logs, validate content, and aggregate and filter data in real-time.

3. It Connects to API Gateway and Other Connection Points:

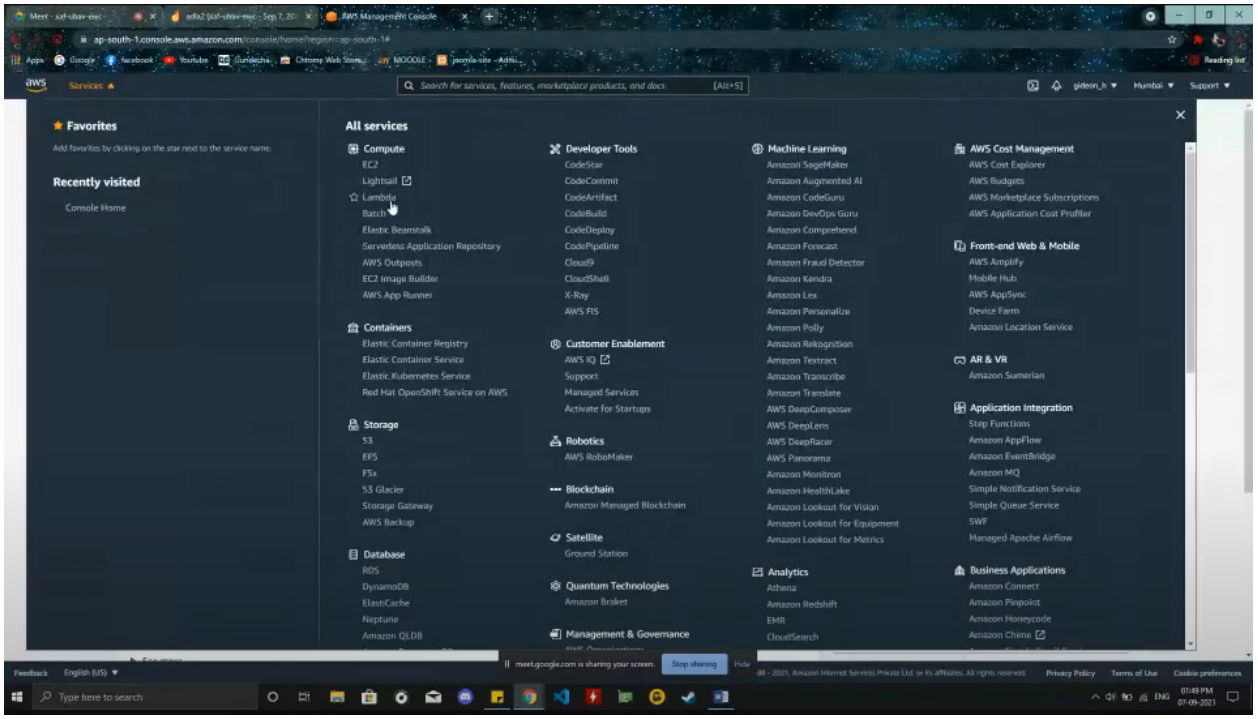
In a manner of speaking, the API Gateway is the portal to your microcosm within AWS. It'll connect to many of the cloud services directly, but when you need flexibility, Lambda is your go-to. See, it's where you put custom code to process requests that come in through the API Gateway. Use a Lambda when you need to access several services or do custom processing.

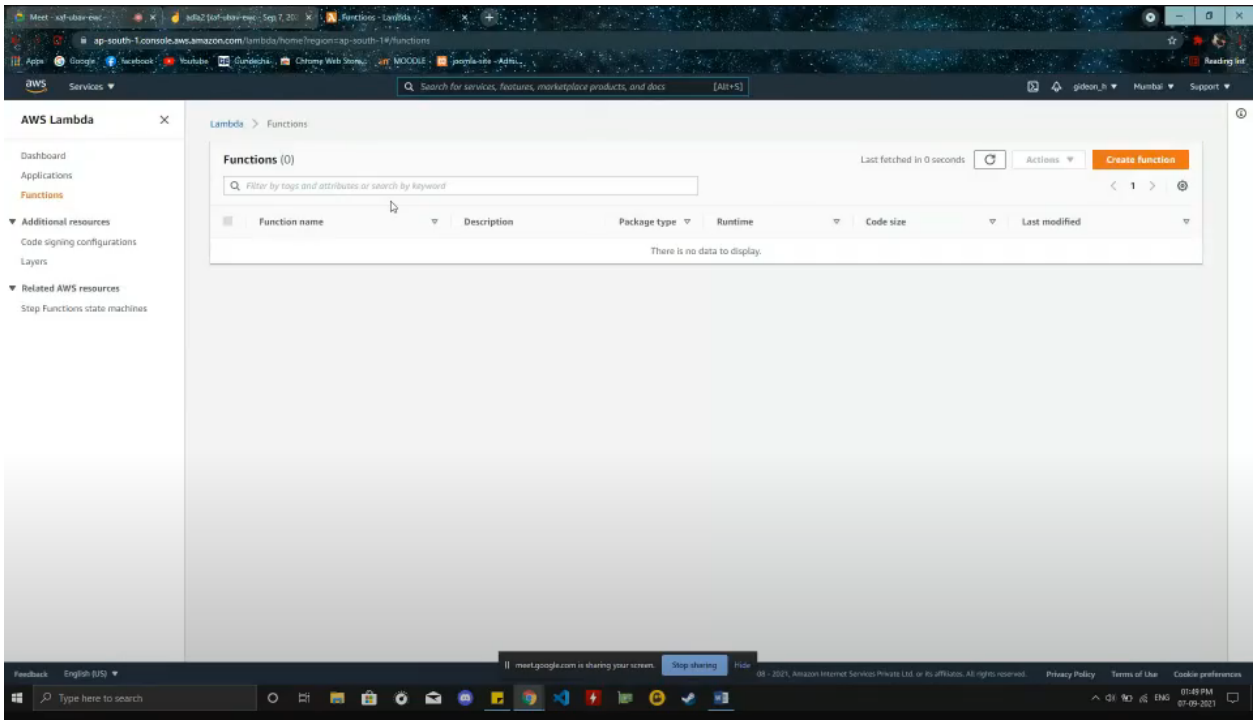
4. You Only Pay for What You Use

As mentioned in the introduction, you'll only pay for what you use. Billing for functions is based on 100 ms intervals.
Interestingly, this creates an incentive to use more efficient code if you're considering the costs. Also, you can tune the maximum resource usage of each Lambda, which affects the base price for your function executions.

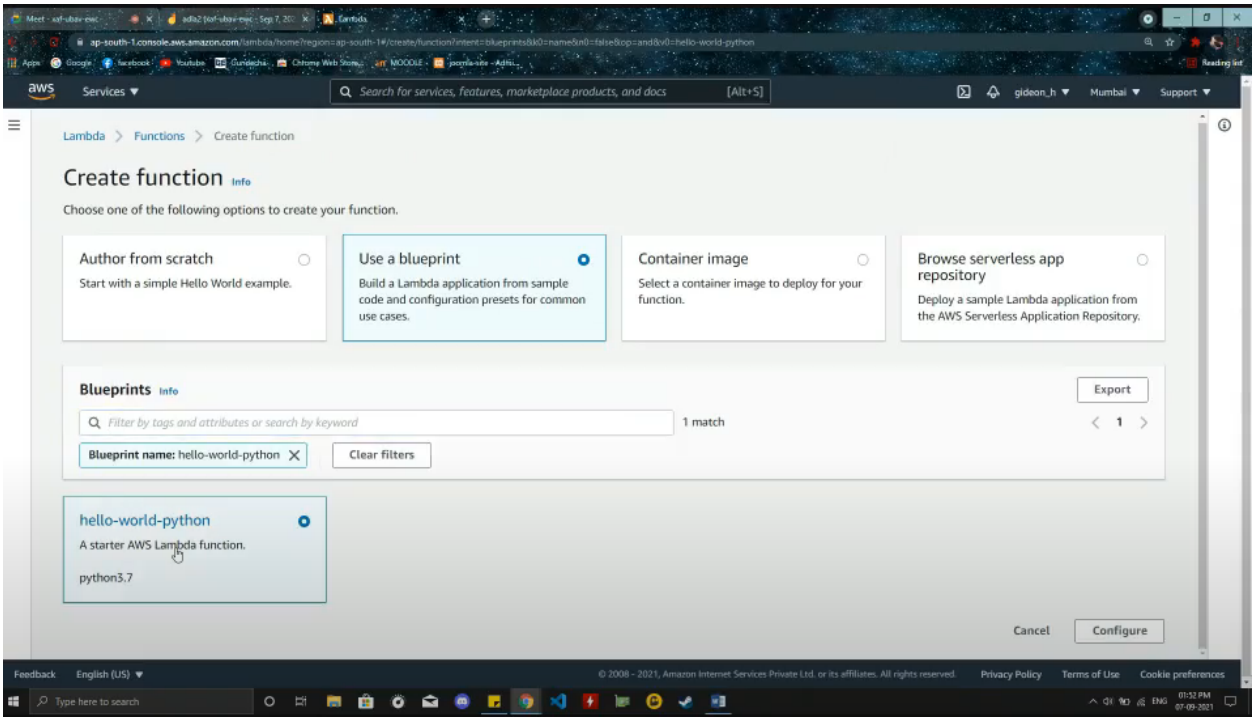
b. Execute following and attach screenshots:

- **Enter Lambda service from console**

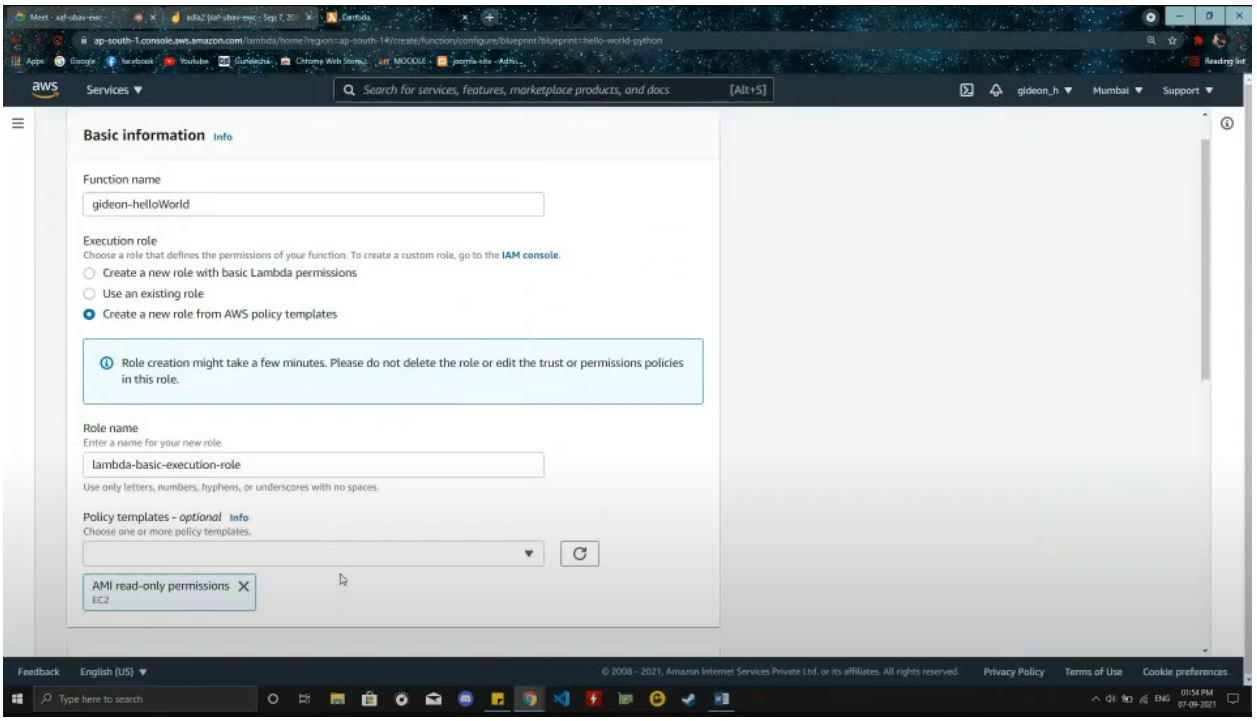




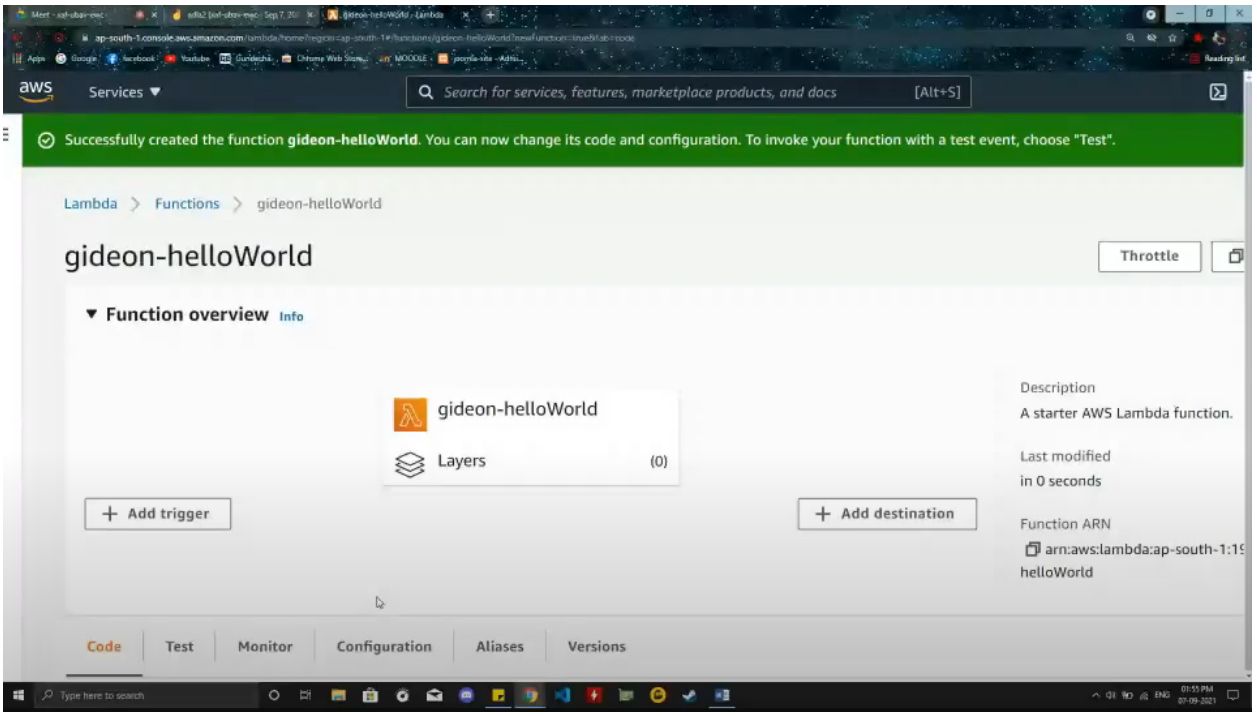
- **Select a Lambda Blueprint**

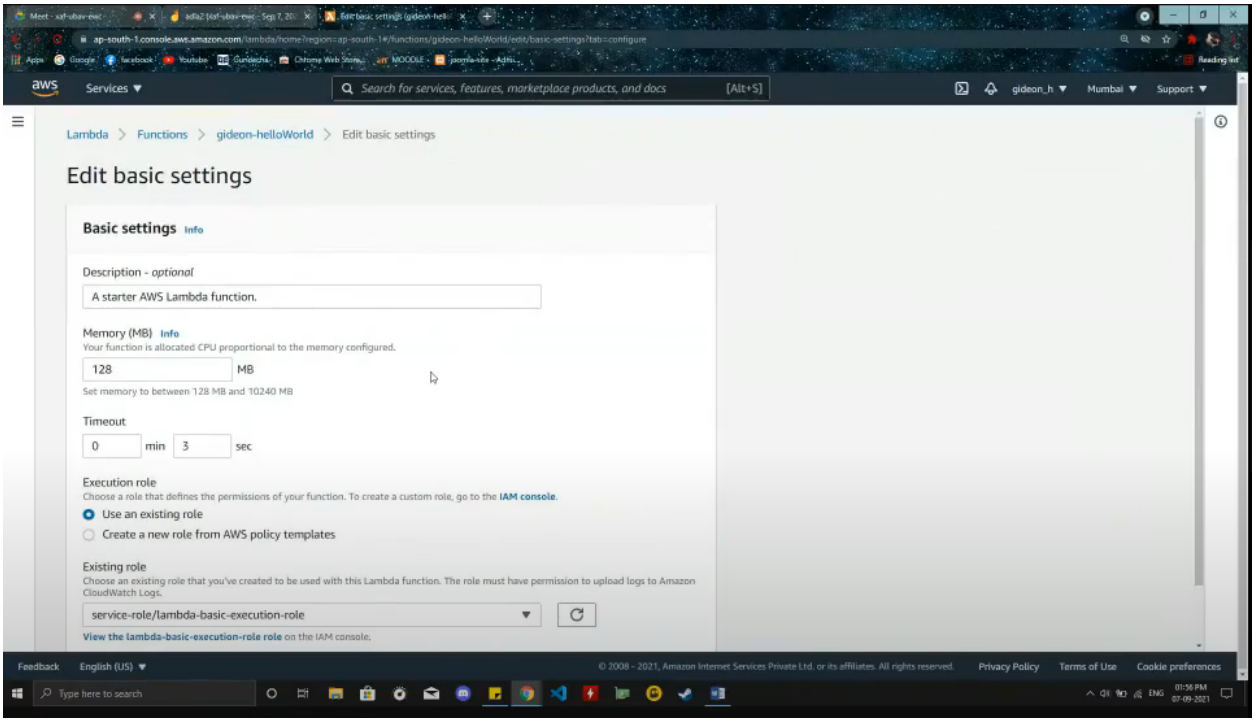


- **Configure and create Lambda function**

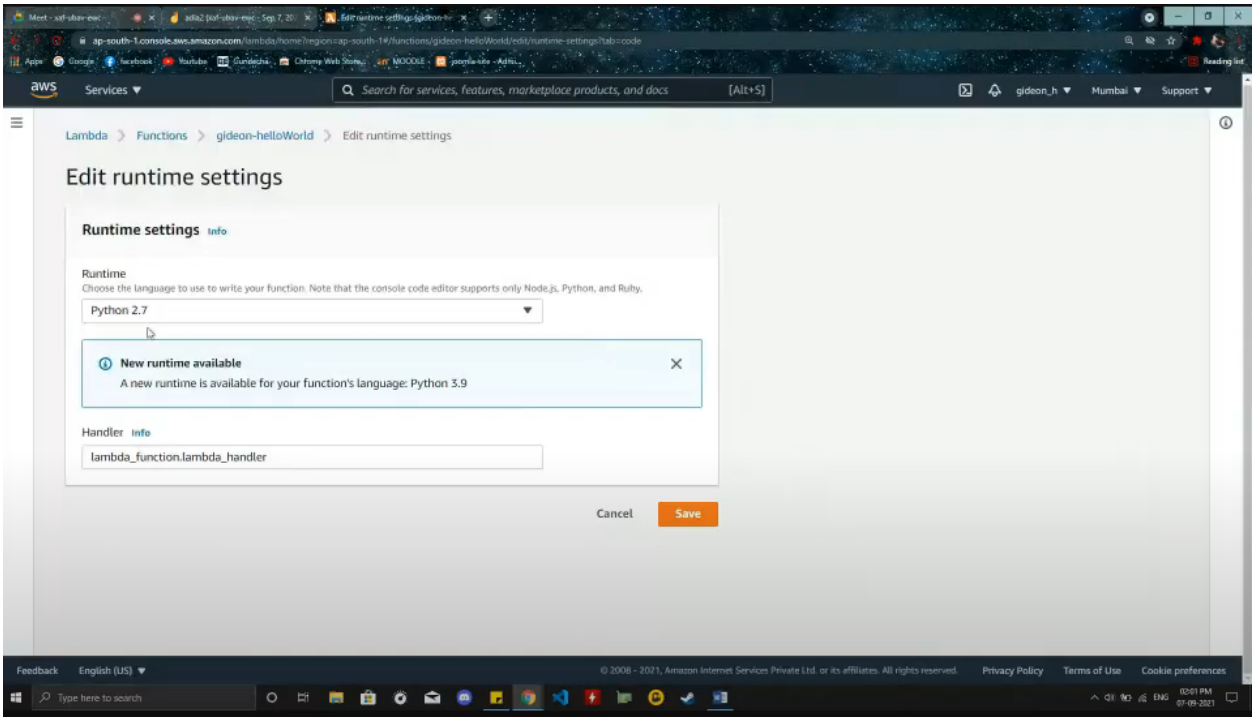


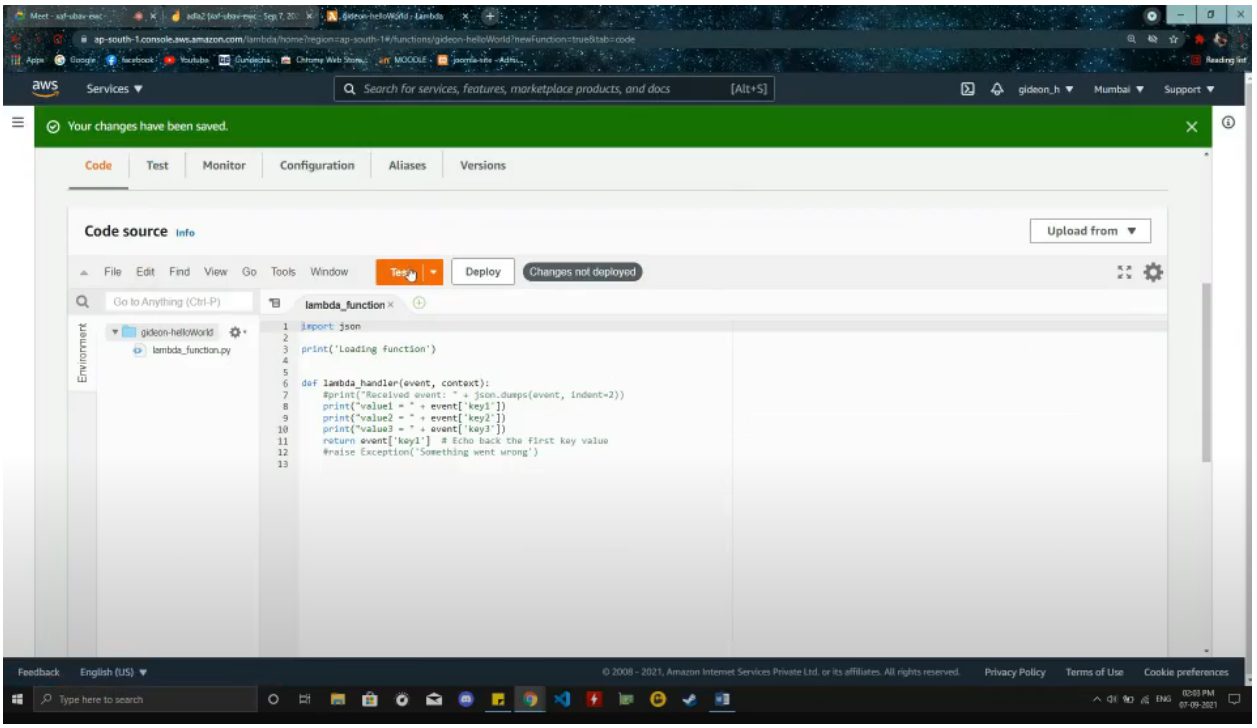
Name: Gideon Harpanahalli **Class**: TEITA **RollNo**: 33 **PID**: 191048 **Batch**: 2



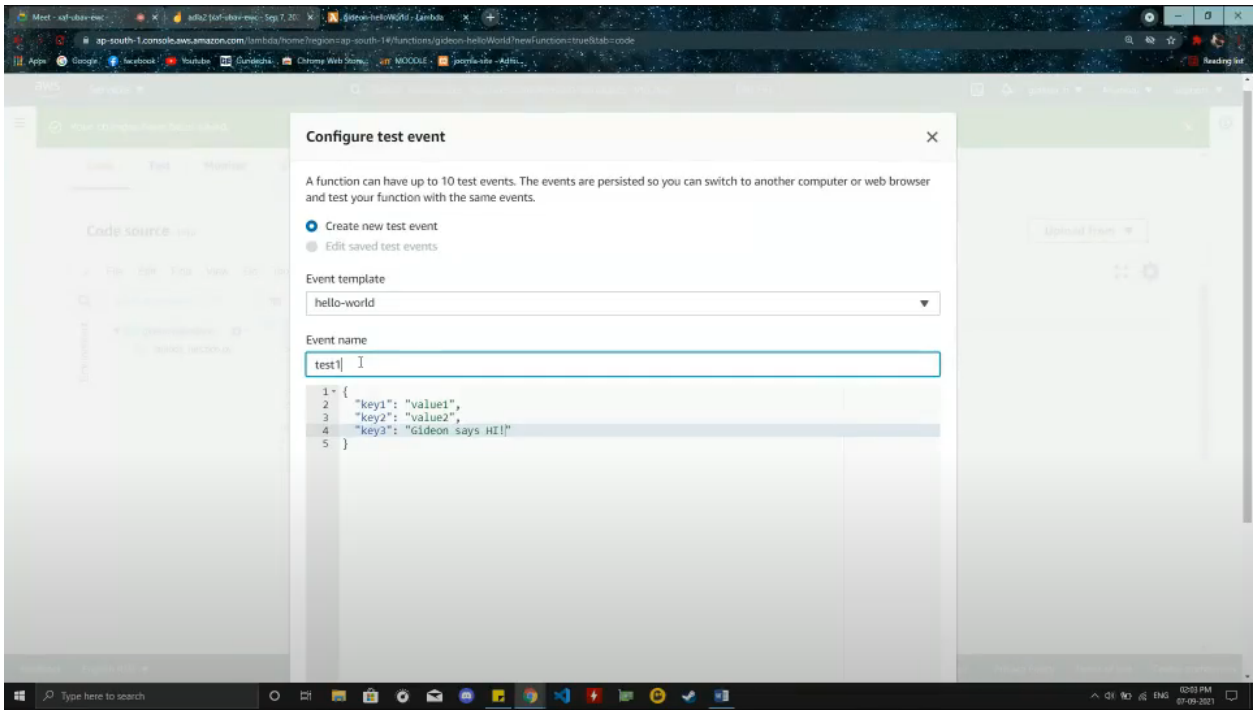


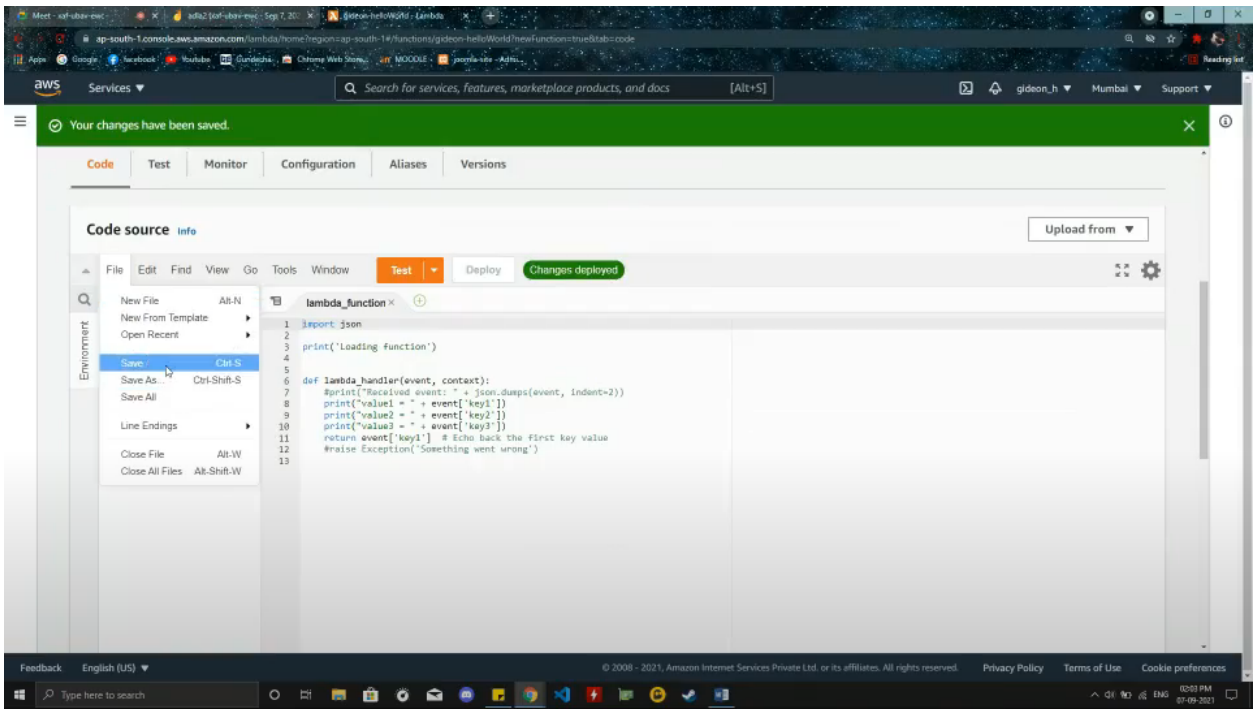
Name: Gideon Harpanahalli **Class**: TEITA **RollNo**: 33 **PID**: 191048 **Batch**: 2



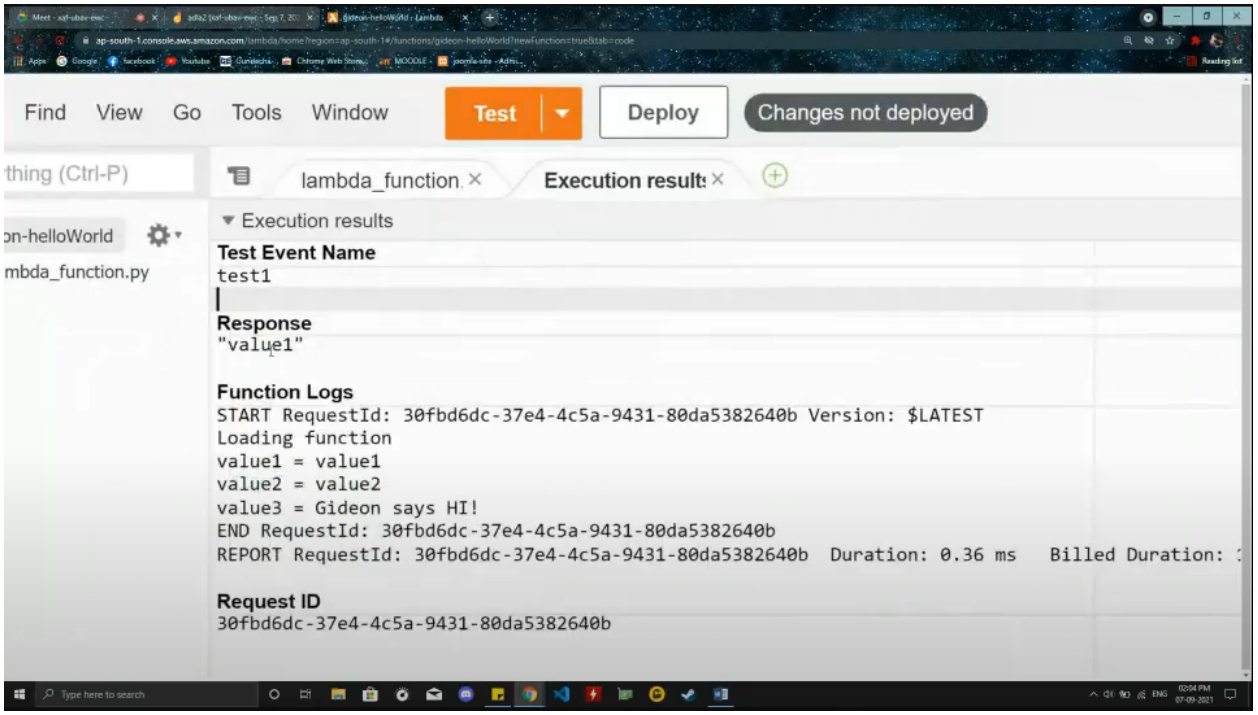


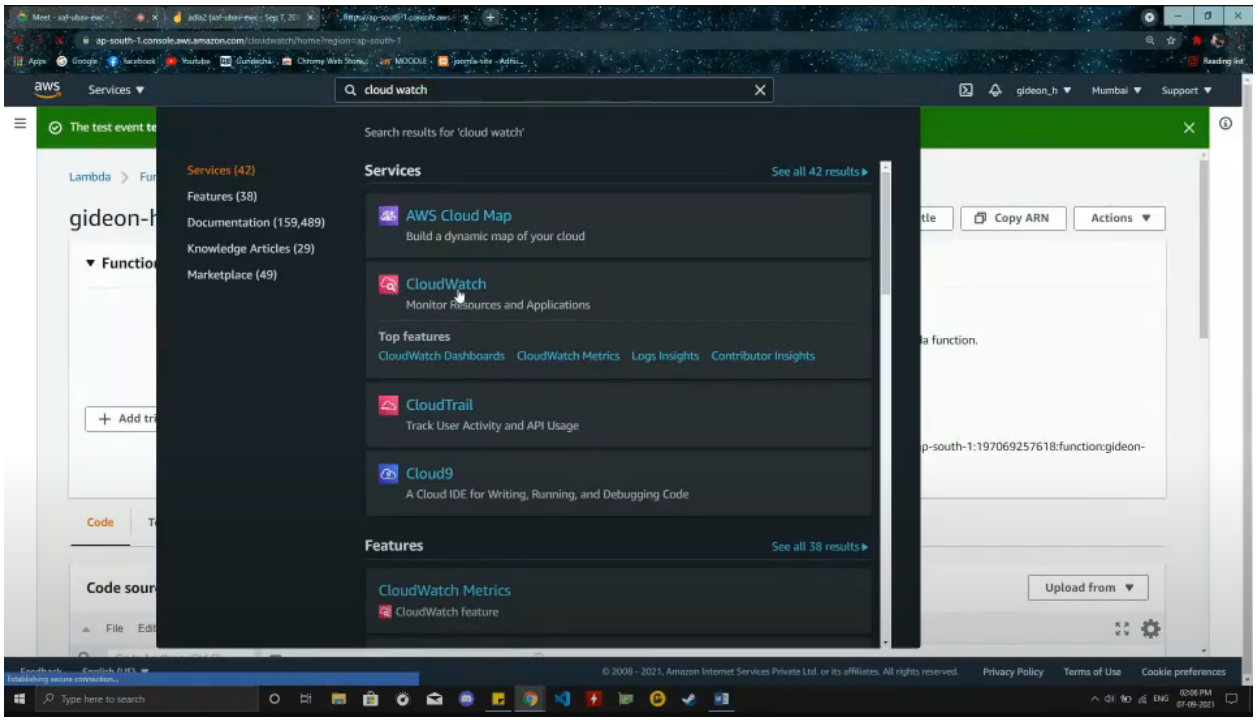
Name: Gideon Harpanahalli **Class**: TEITA **RollNo**: 33 **PID**: 191048 **Batch**: 2



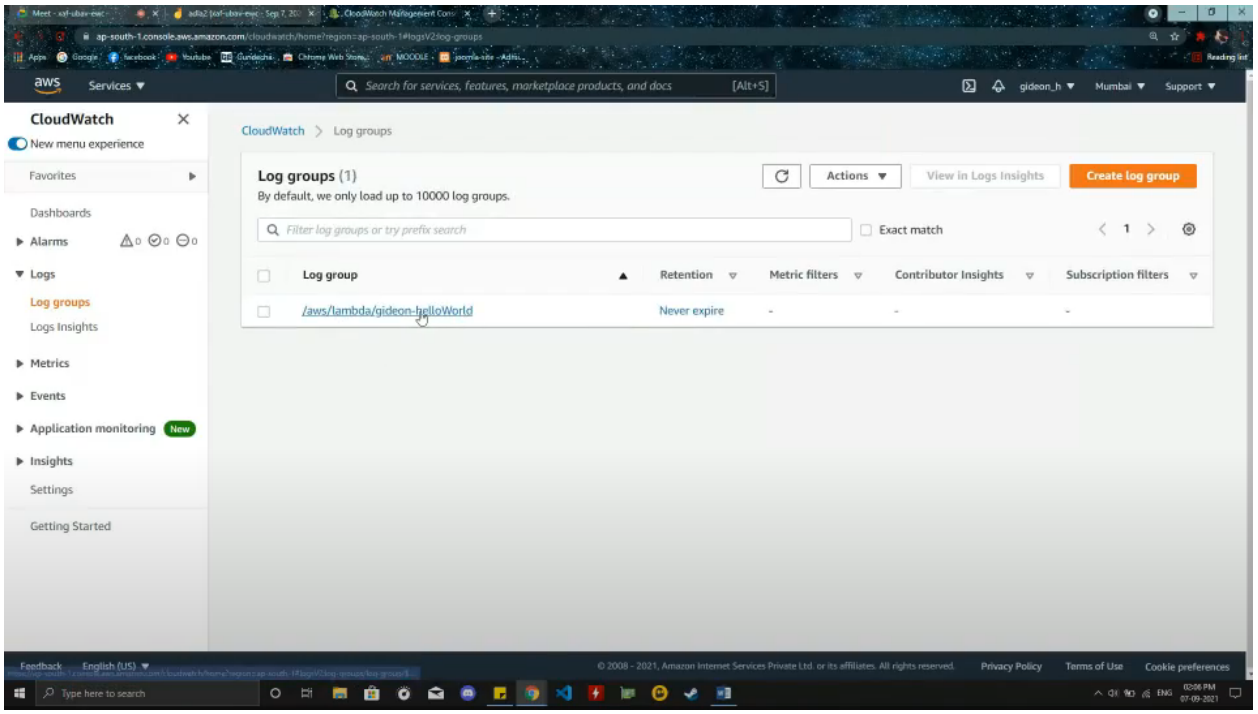


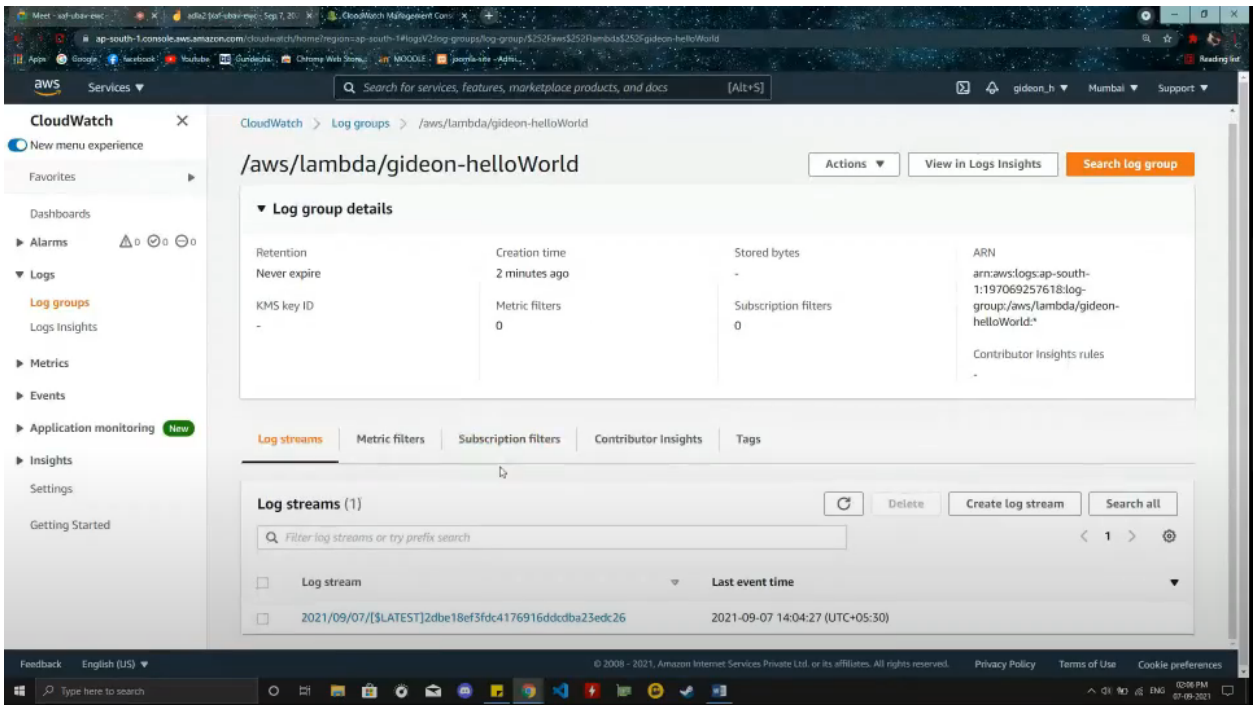
- **Invoke Lambda function and verify results**



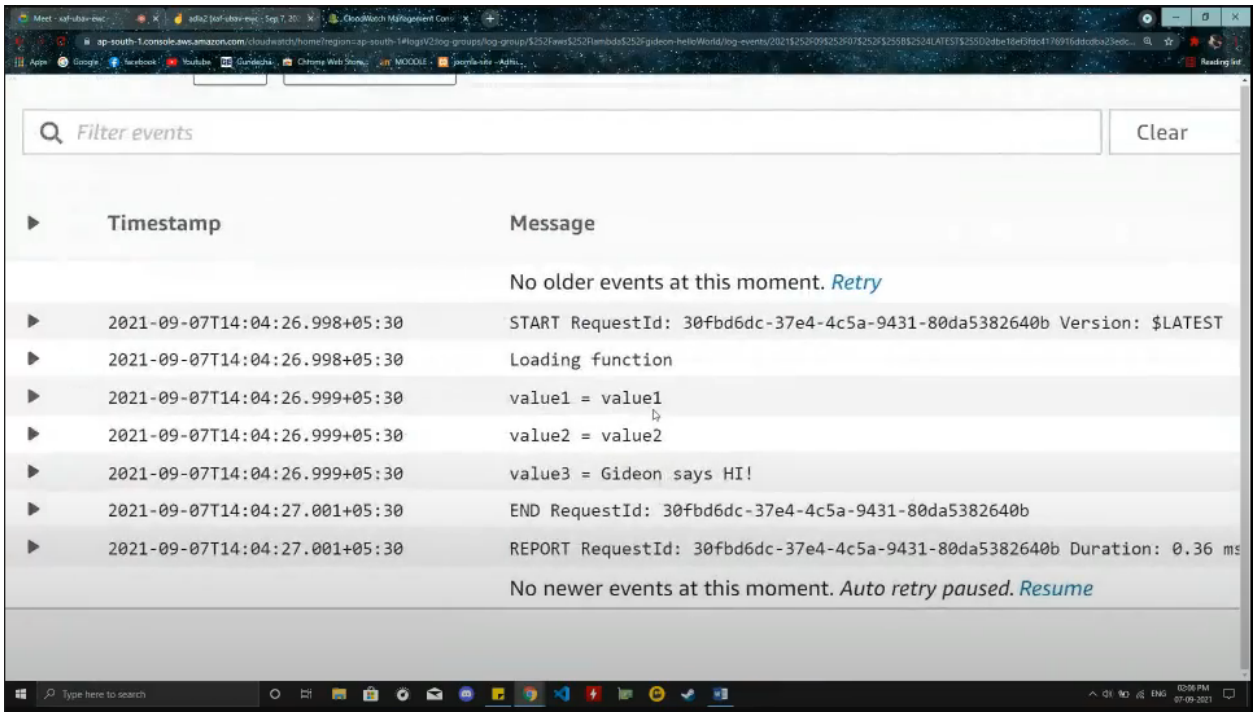


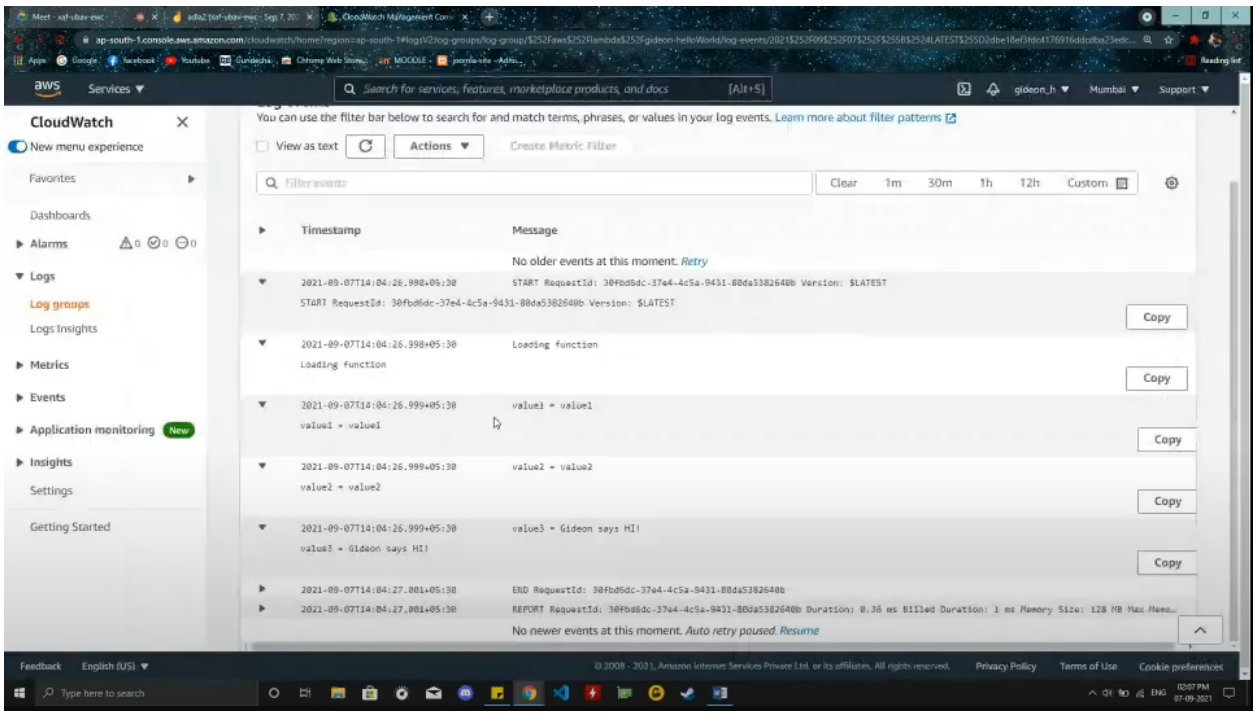
Name: Gideon Harpanahalli **Class**: TEITA **RollNo**: 33 **PID**: 191048 **Batch**: 2





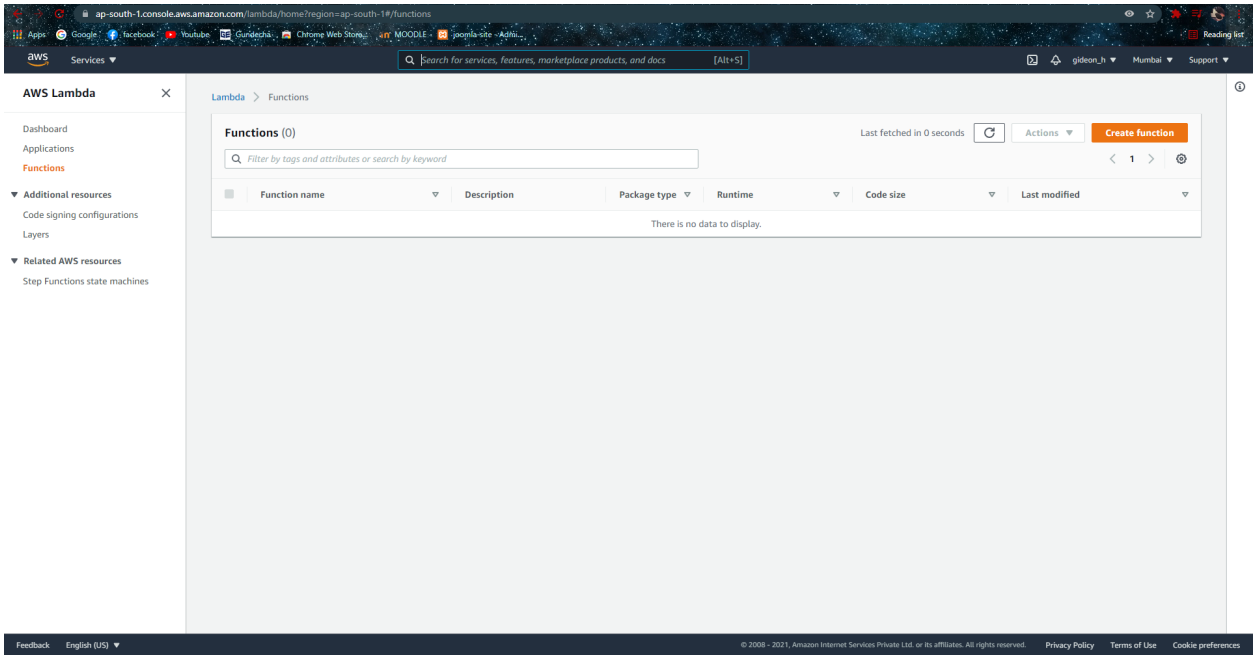
Name: Gideon Harpanahalli **Class**: TEITA **RollNo**: 33 **PID**: 191048 **Batch**: 2





- **Clean up the lambda Function and free all the resources.**

(lambda function cleared)



8. Post-Experiments Exercise

A. Extended Theory:

Create the Lambda function using Node.js

→ **To create an execution role**

1. Open the roles page in the IAM console.
2. Choose Create role.
3. Create a role with the following properties.
   - Trusted entity – Lambda.
   - Permissions – AWSLambdaBasicExecutionRole.
   - Role name – lambda-role.

   The AWSLambdaBasicExecutionRole policy has the permissions that the function needs to write logs to CloudWatch Logs.

   You can add permissions to the role later, or swap it out for a different role that's specific to a single function.

To create a Node.js function

1. Open the Lambda console.
2. Choose Create function.
3. Configure the following settings:
   - Name – my-function.
   - Runtime – Node.js 14.x.
   - Role – Choose an existing role.
   - Existing role – lambda-role.

4. Choose Create function.
5. To configure a test event, choose Test.
6. For Event name, enter test.
7. Choose Save changes.
8. To invoke the function, choose Test.

The console creates a Lambda function with a single source file named index.js. You can edit this file and add more files in the built-in code editor. To save your changes, choose Save. Then, to run your code, choose Test.

The index.js file exports a function named handler that takes an event object and a context object. This is the handler function that Lambda calls when the function is invoked. The Node.js function runtime gets invocation events from Lambda and passes them to the handler. In the function configuration, the handler value is index.handler.

When you save your function code, the Lambda console creates a .zip file archive deployment package. When you develop your function code outside of the console (using an SDE) you need to create a deployment package to upload your code to the Lambda function.

The function runtime passes a context object to the handler, in addition to the invocation event. The context object contains additional information about the invocation, the function, and the execution environment. More information is available from environment variables.

Your Lambda function comes with a CloudWatch Logs log group. The function runtime sends details about each invocation to CloudWatch Logs. It relays any logs that your function outputs during invocation. If your function returns an error, Lambda formats the error and returns it to the invoker

B. Questions:

GIDEON HARPANAHALLI

TEITA [33] ADL Exp 11

DOMS Page No. Date / /

B] Questions

- What is AWS Lambda Function.

→ AWS lambda is a serverless Compute service that runs code in response to a trigger.

→ Automatically manages the underlying compute resources.

→ AWS lambda can extend othe AWS services with custom logic or create your own backend services that operate at AWS scale, performance.

→ AWS lambda can run code in response to HTTP request etc. via Amazon API Gateway

→ modifications to objects in Amazon S3 buckets, table updates in Amazon Dynamo DB & state transitions in AWS Step functions.

- EC2 Lambda vs Elastic Beanstalk

→ Elastic Beanstalk Service is fast in deployment & managing Apps in cloud.

→ Beanstalk automatically handles the deployment details of capacity, load balancing, auto-scaling & application health monitoring.

→ Lambda service directly starts running code responding to modifications that are made to objects, updates or messages.

→ AWS Lambda goes on its own in managing the underlying infrastructure for your compute resources.

→ EC2 we need to operate with virtual machines (VMs) known as EC2 instances.

→ Needs pre configuration of image with an installed OS and create customized AMI.

C. Conclusion:

GIDEON HARPANAHALLI
TEITA [33] ADL EXP 11

2015 Page No.
Date / /

C] CONCLUSIONS

(1)

→ This experiment dealt with AWS Lambda functions and its workflows, understood serverless compute.
→ We created a lambda function using python 2.7 tested it.
→ We made changes in the default code and displayed result.
→ Cloudwatch log ~~session~~ section showed the result, along with time stamps

(2)

→ Lambda function is one of the most effective way to deploy and run code in cloud, in terms of cost and ease of deployment.
→ Lambda function automatically manages and provisions the infrastructure for the developer, thus reducing the ~~toa~~ administrative load on the developer.
→ Highly flexible and integrates well with other AWS services.

D. References:

https://aws.amazon.com/getting-started/hands-on
https://www.scalyr.com/blog/aws-lambda-tutorial/