Run a Serverless "Hello, World!"

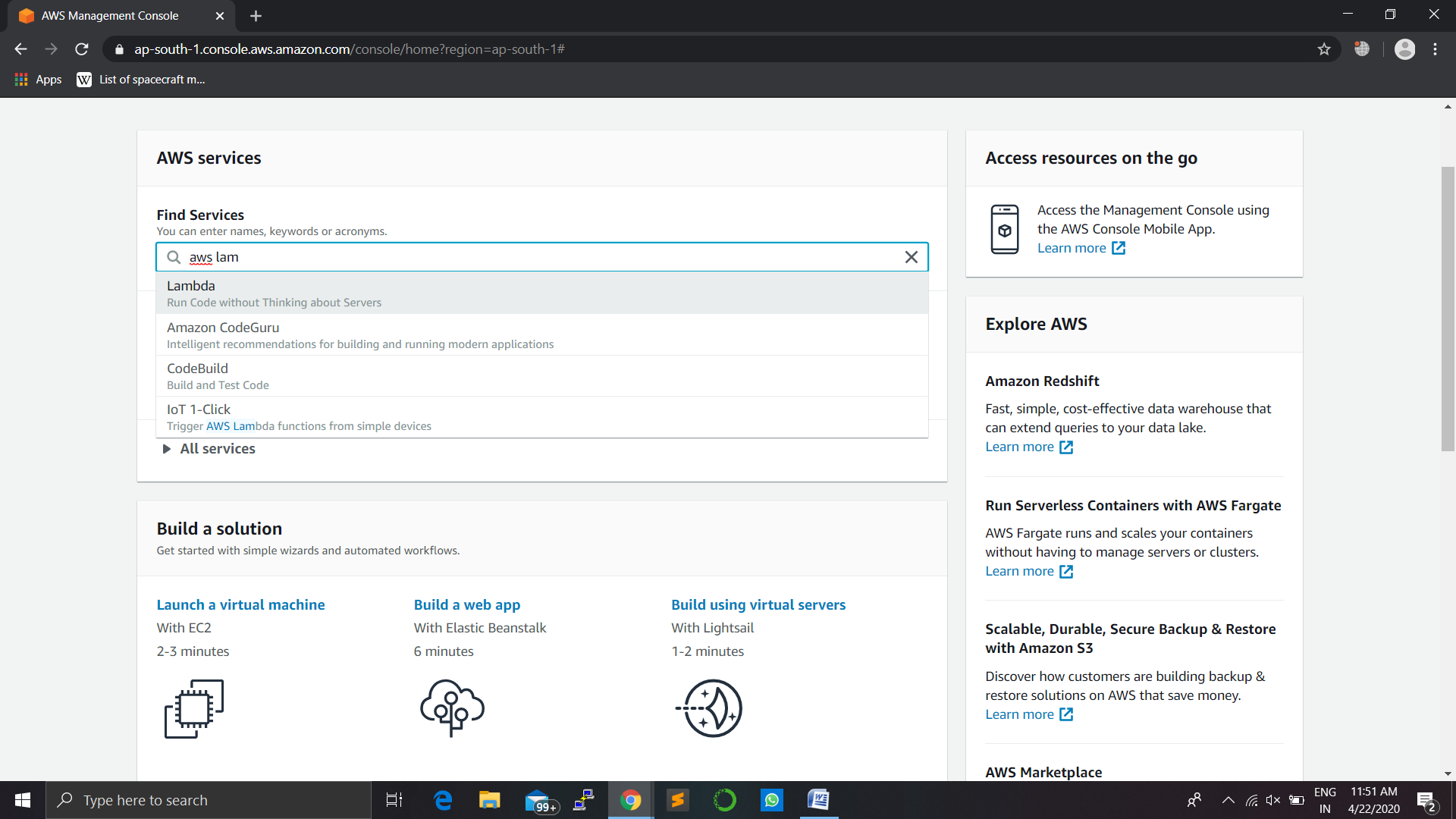
with AWS Lambda

In this we will learn the basics of running code on AWS Lambda without provisioning or managing servers. We will walk through how to create a Hello World Lambda function using the AWS Lambda console. We will then show you how to manually invoke the Lambda function using sample event data and review your output metrics.

Everything done in this tutorial is free tier eligible.

## Step 1: Enter the Lambda Console

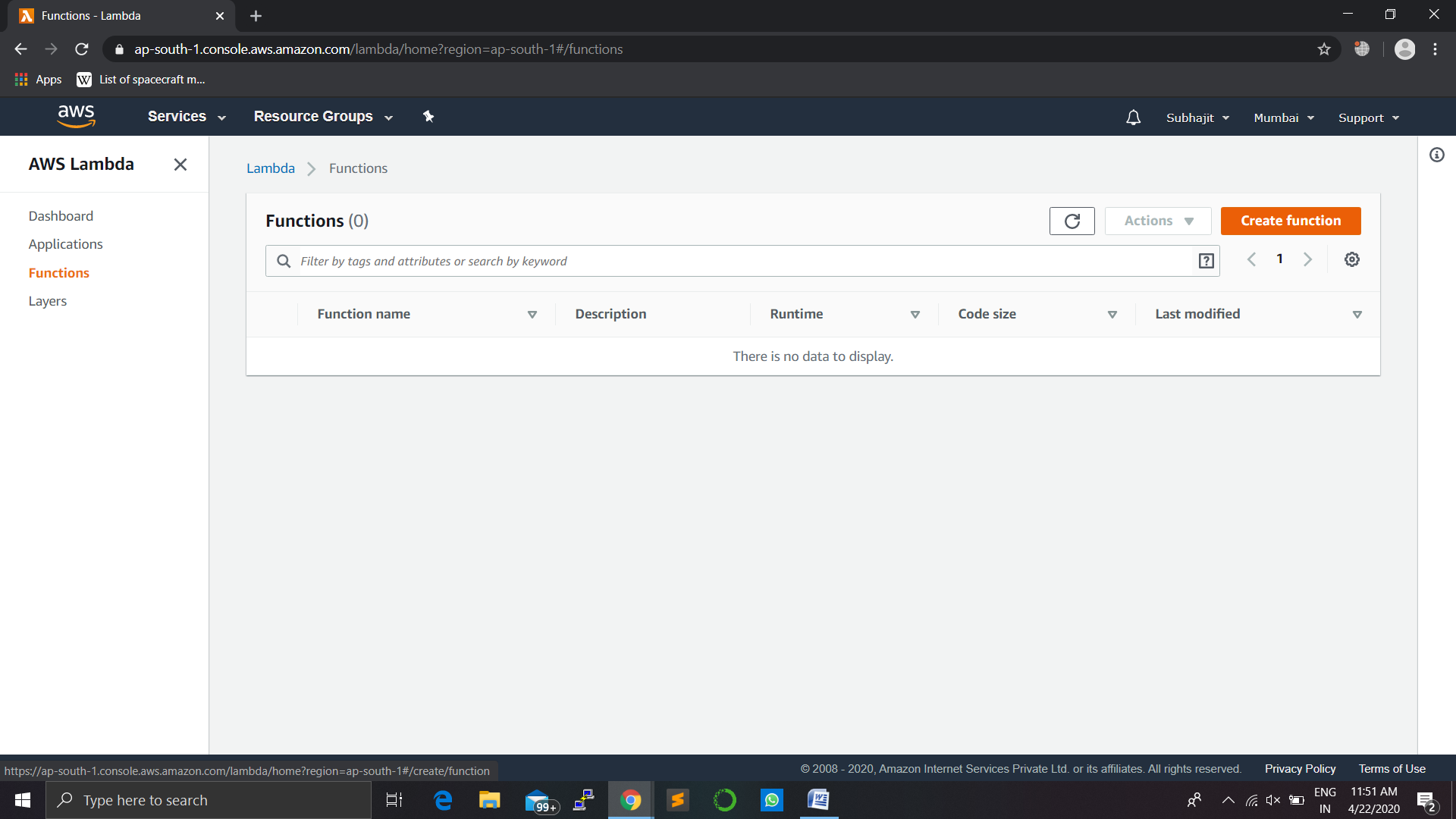
When you [click here](https://console.aws.amazon.com/console/home), the AWS Management Console will open in a new browser window, so you can keep this step-by-step guide open.  Find Lambda under *Compute*and click to open the AWS Lambda Console.



## Step 2: Select a Lambda Blueprint

Blueprints provide example code to do some minimal processing. Most blueprints process events from specific event sources, such as Amazon S3, DynamoDB, or a custom application.

a.  In the AWS Lambda console, select Create a Function.

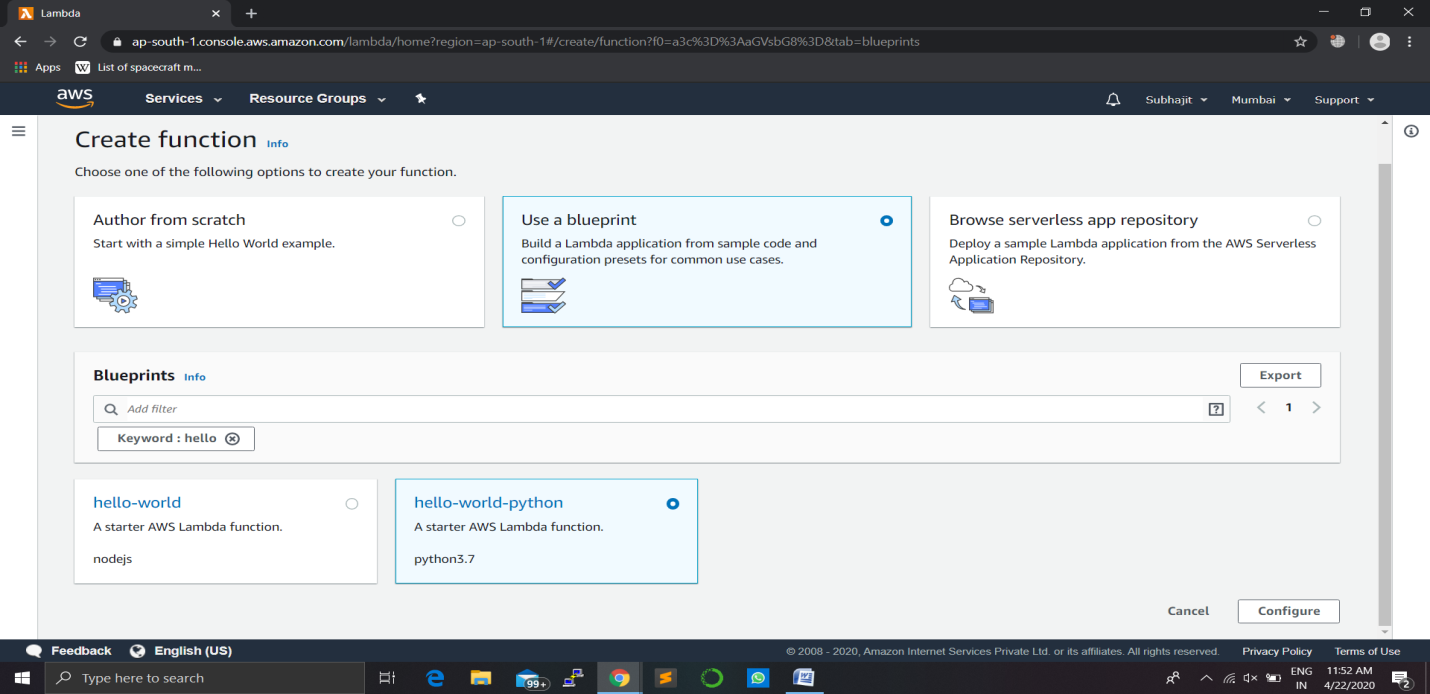


Note:  
The console shows this page only if you do not have any Lambda functions created. If you have created functions already, you will see the *Lambda > Functions* page. On the list page, choose *Create a function* to go to the Create function page.

b.  Select Blueprints.

c.  In the Filter box, type in *hello-world-python*and select the hello-world-python blueprint.

d.  Then click Configure.



## Step 3: Configure and Create Your Lambda Function

A Lambda function consists of code you provide, associated dependencies, and configuration. The configuration information you provide includes the compute resources you want to allocate (for example, memory), execution timeout, and an IAM role that AWS Lambda can assume to execute your Lambda function on your behalf.

a.  You will now enter *Basic Information*about your Lambda function.

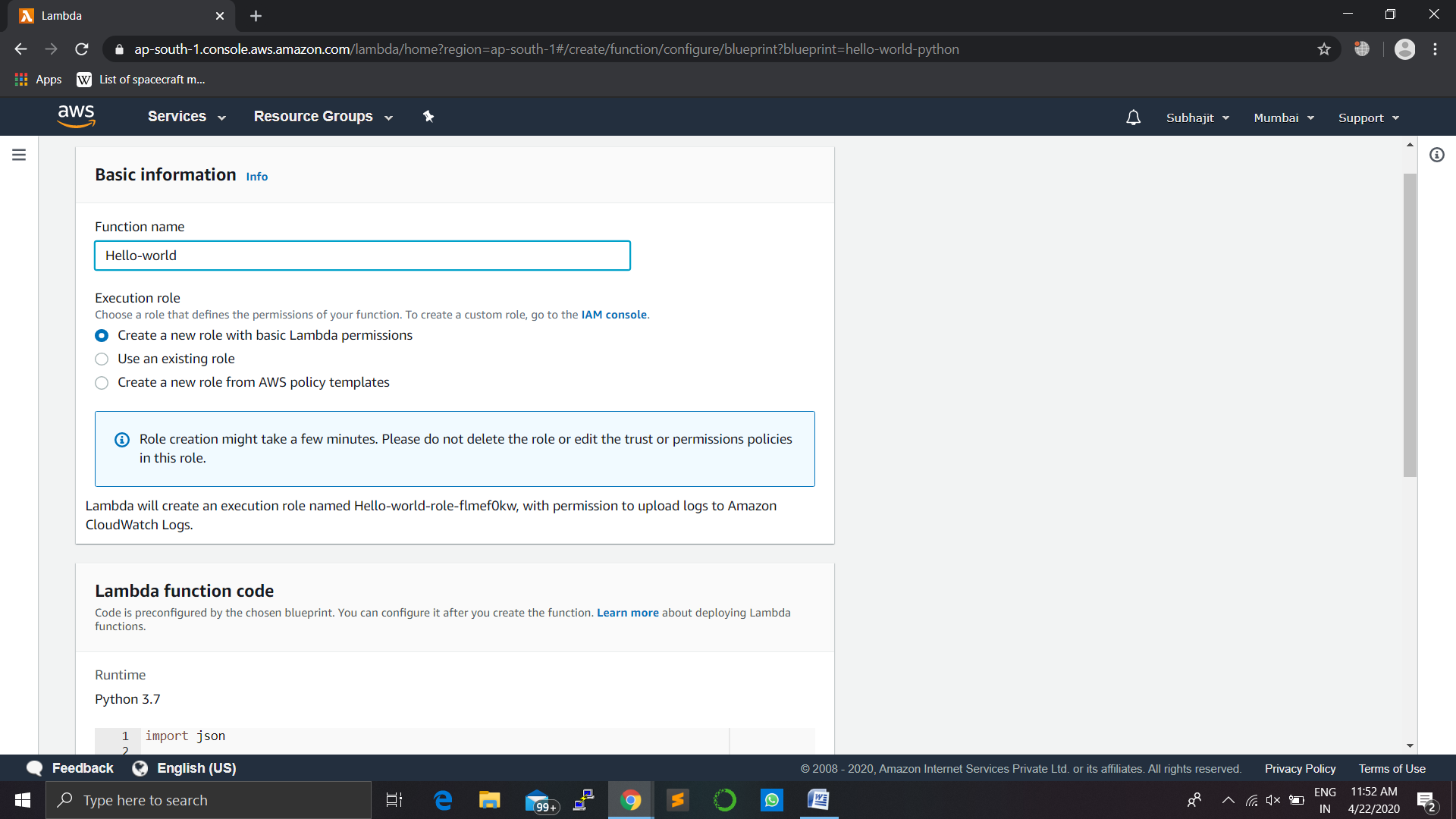
Basic Information:

* Name: You can name your Lambda function here. For this tutorial, enter *hello-world-python*.
* Role: You will create an IAM role (referred as the execution role) with the necessary permissions that AWS Lambda can assume to invoke your Lambda function on your behalf. Select *Create new role from template(s).*
* Role name: type *lambda\_basic\_execution*

Lambda Function Code:

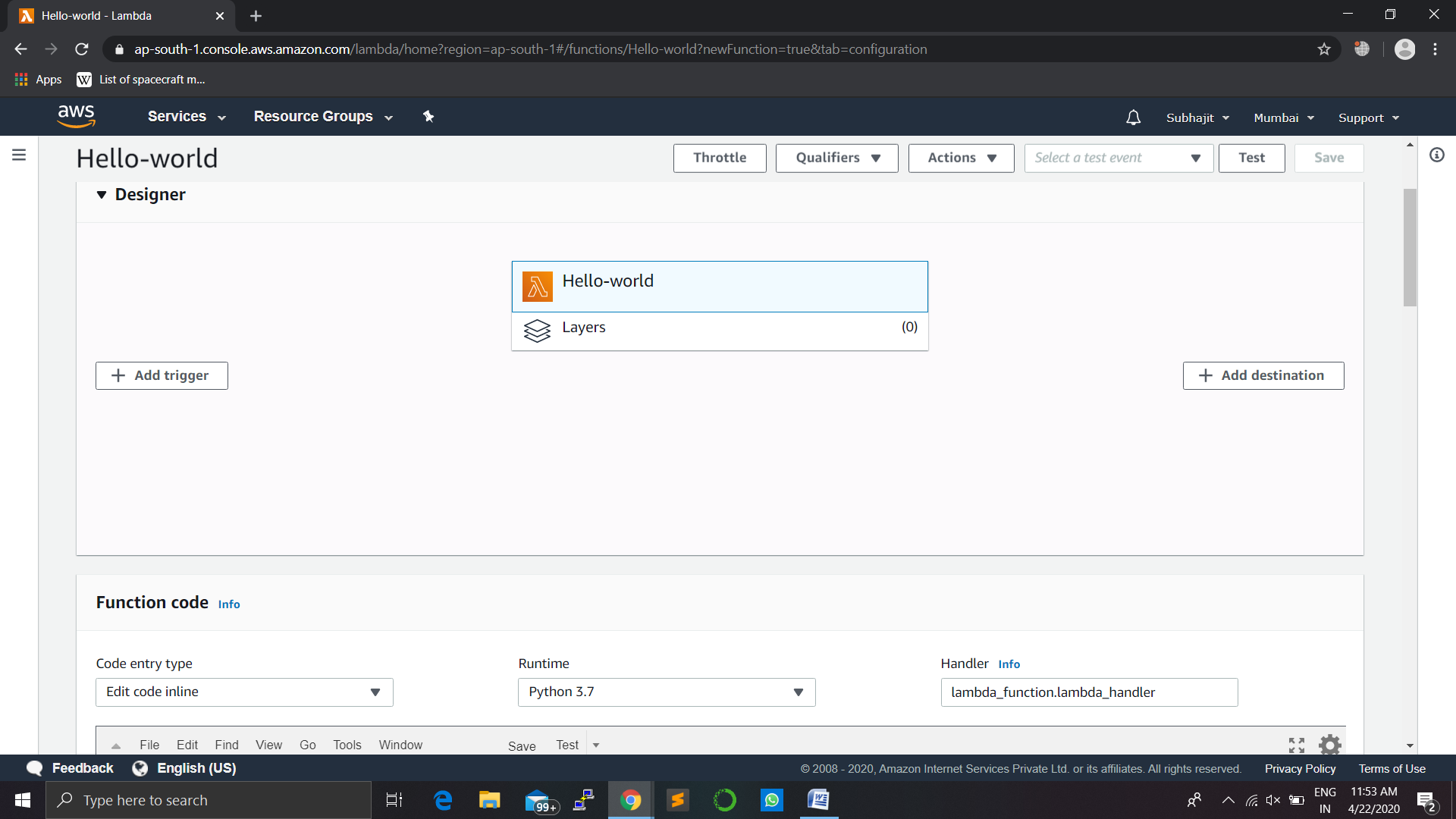
* In this section, you can review the example code authored in Python.

b.  Go to the bottom of the page and select Create Function.

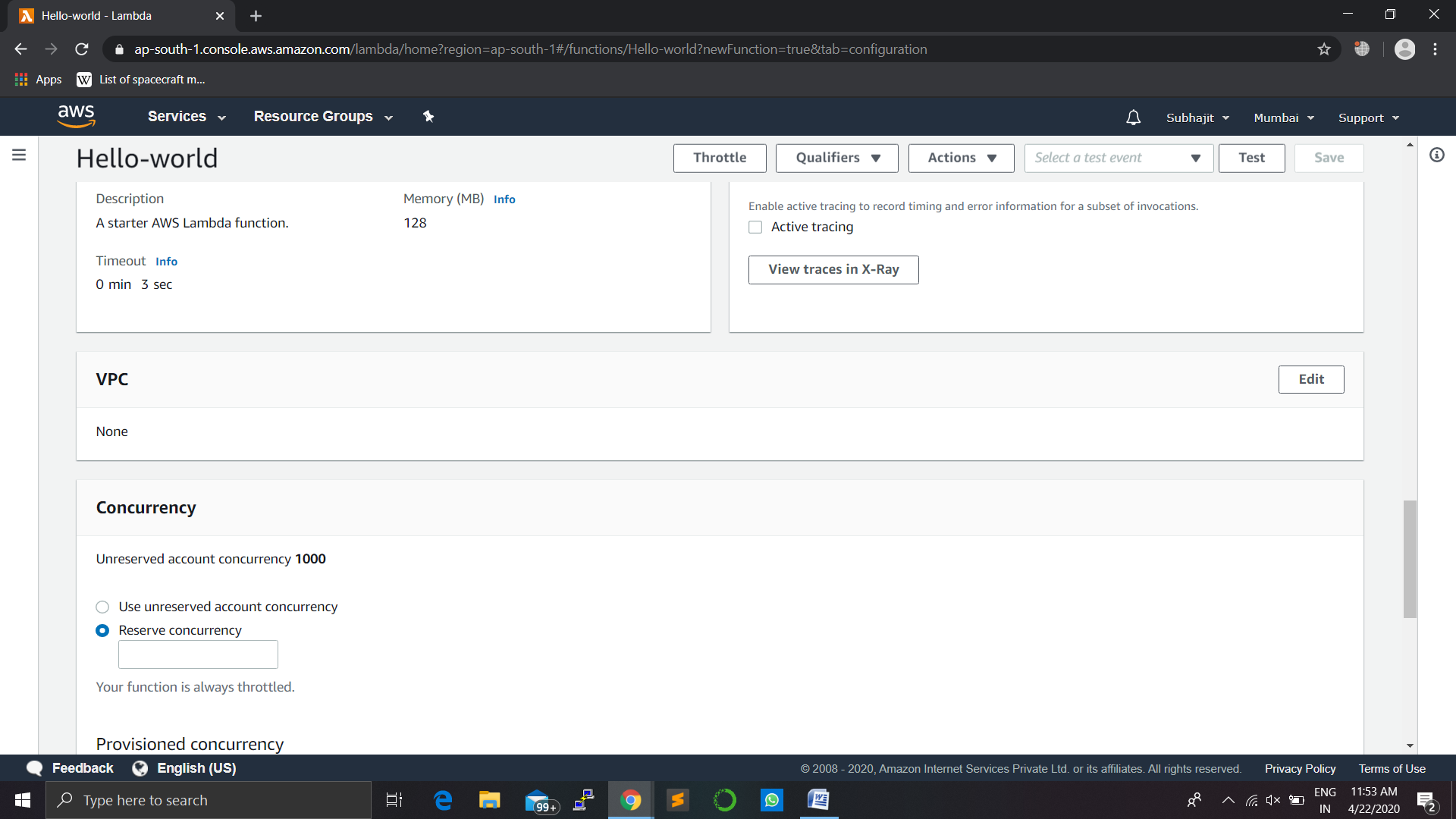


c.  Runtime: Currently, you can author your Lambda function code in Java, Node.js, C#, Go or Python. For this tutorial, leave this on *Python 2.7* as the runtime.

d.  Handler: You can specify a handler (a method/function in your code) where AWS Lambda can begin executing your code. AWS Lambda provides event data as input to this handler, which processes the event.In this example, Lambda identifies this from the code sample and this should be pre-populated with *lambda\_function.lambda\_handler*.



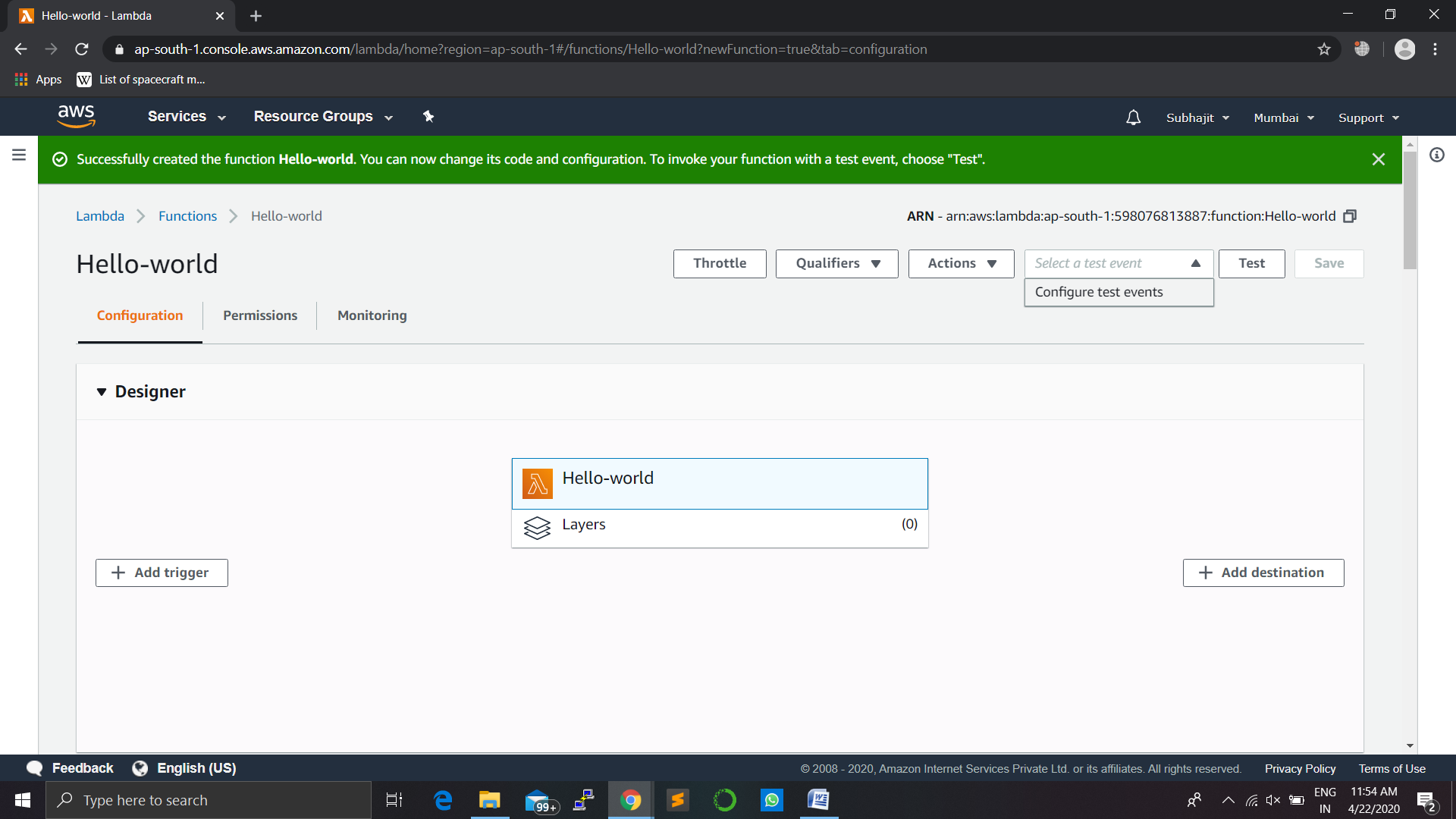
e.  Scroll down to configure your memory, timeout, and VPC settings.  For this tutorial, leave the default Lambda function configuration values.



## Step 4: Invoke Lambda Function and Verify Results

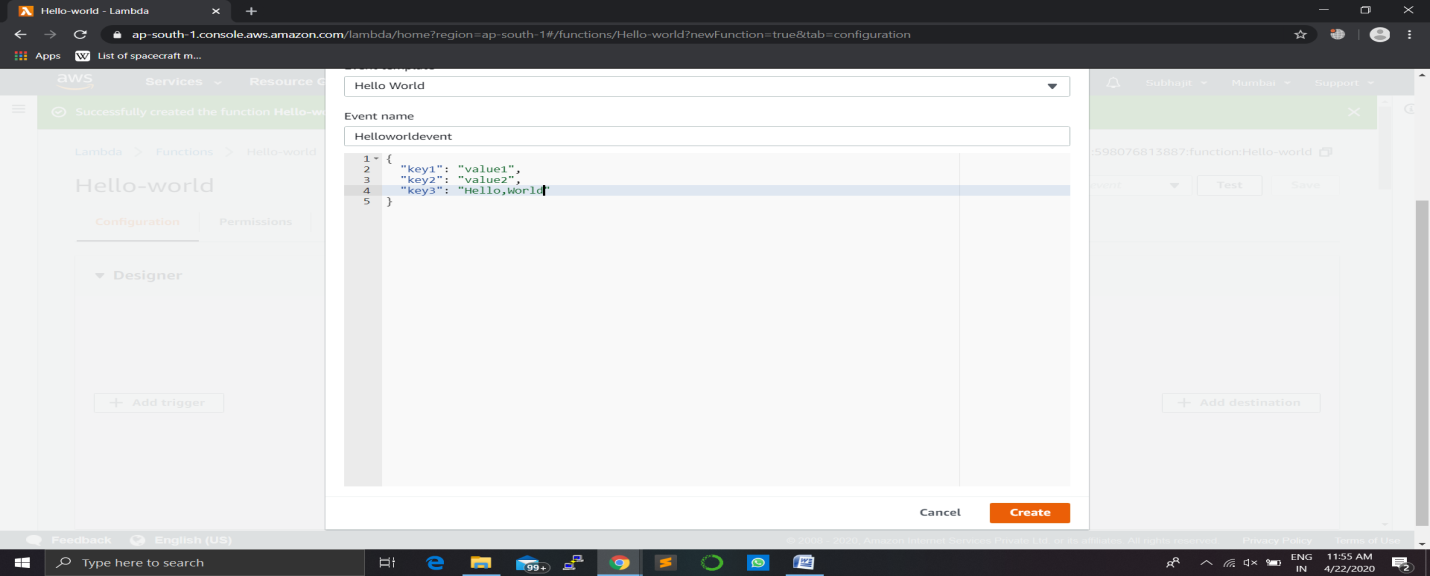
The console shows the hello-world-python Lambda function - you can now test the function, verify results, and review the logs.

a.  Select Configure Test Event from the drop-down menu called "Select a test event...".

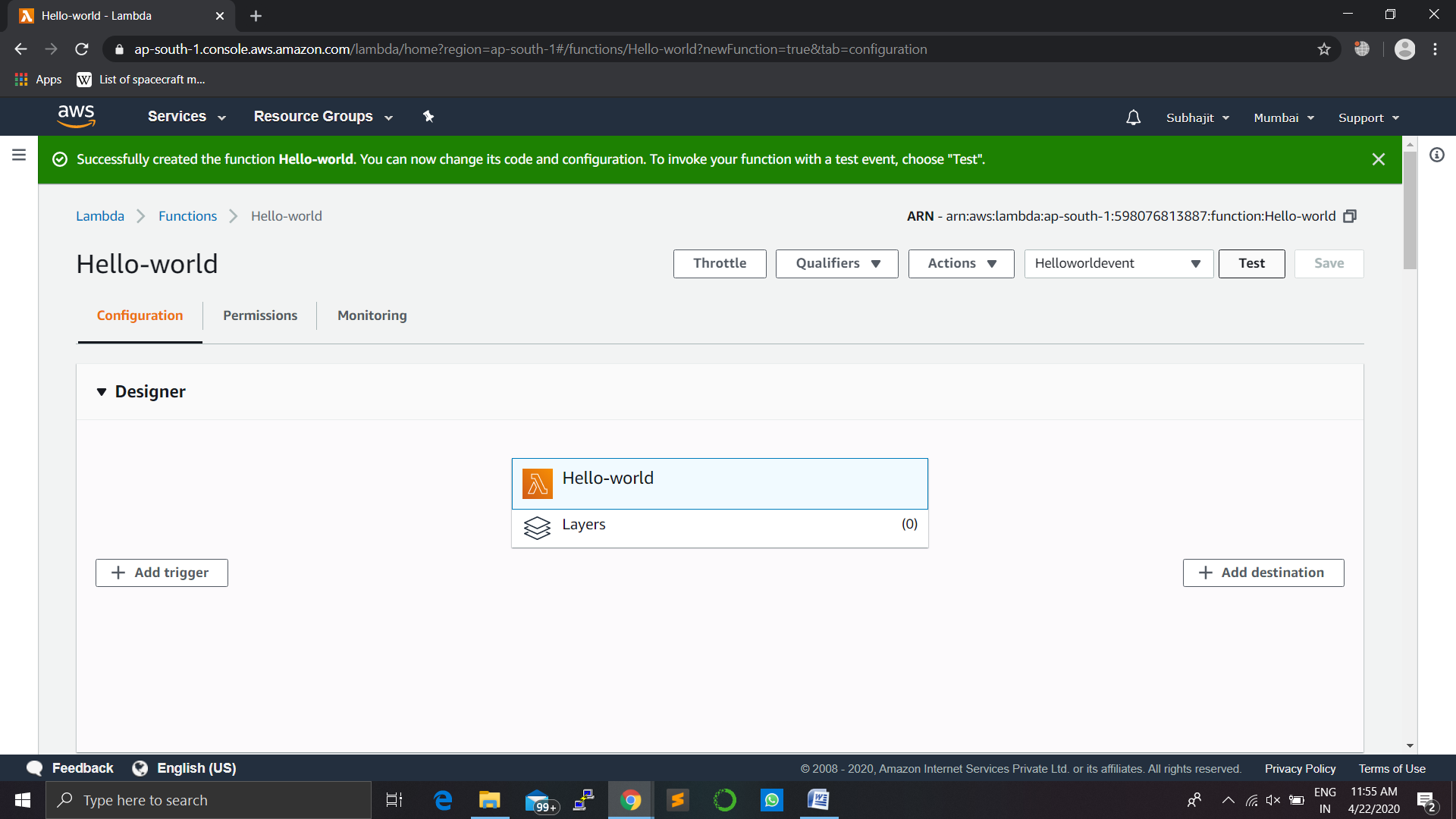


b.  The editor pops up to enter an event to test your function.

* Choose *Hello World* from the Sample event template list from the Input test event page.
* Type in an event name like *HelloWorldEvent.*
* You can change the values in the sample JSON, but don’t change the event structure. For this tutorial, replace *value1*with *hello, world!.*Select Create.

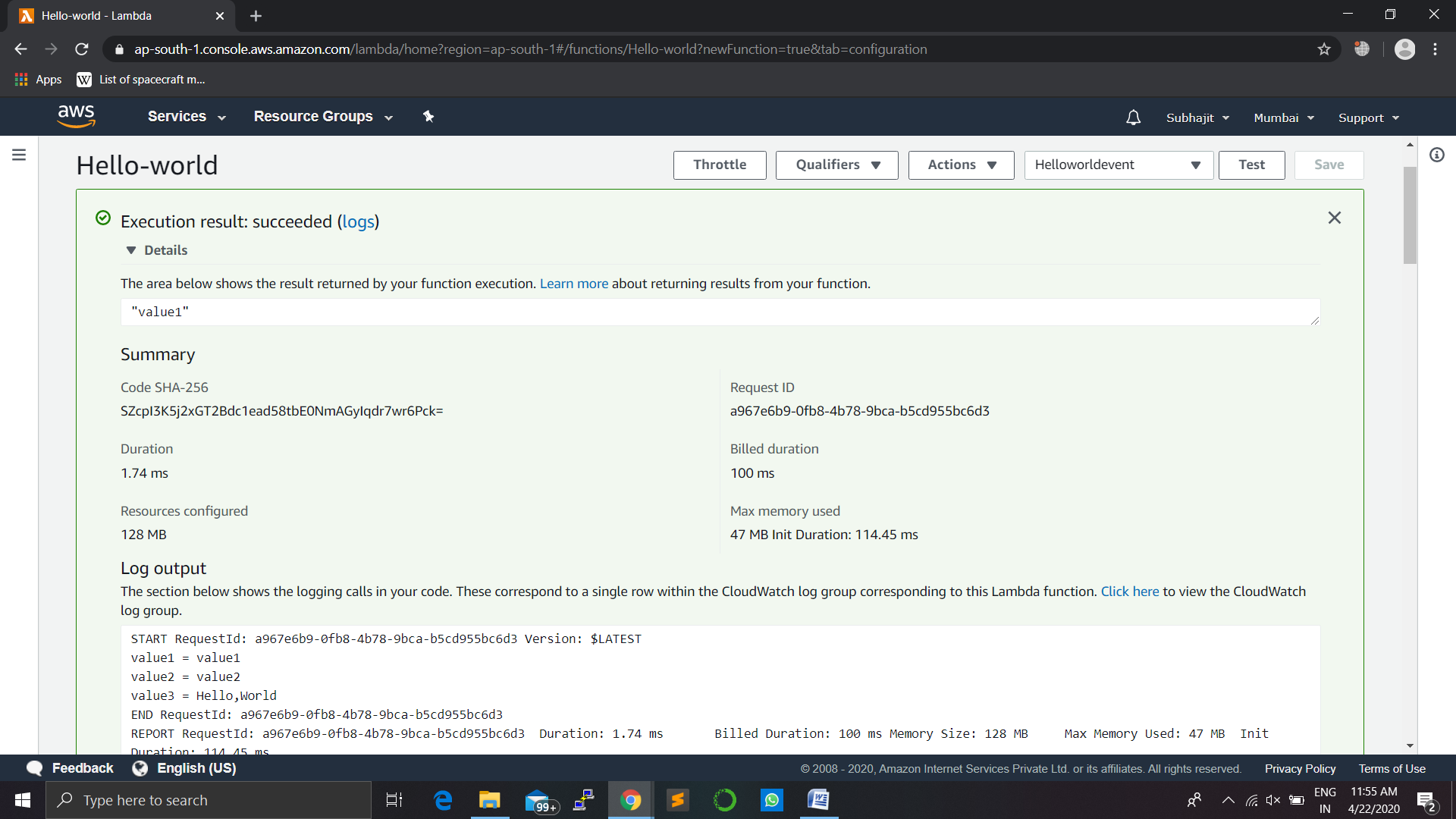


c.  Select Test.



d.  Upon successful execution, view the results in the console:

* The Execution results section verifies that the execution succeeded.
* The Summary section shows the key information reported in the Log output.
* The Log output section will show the logs generated by the Lambda function execution.

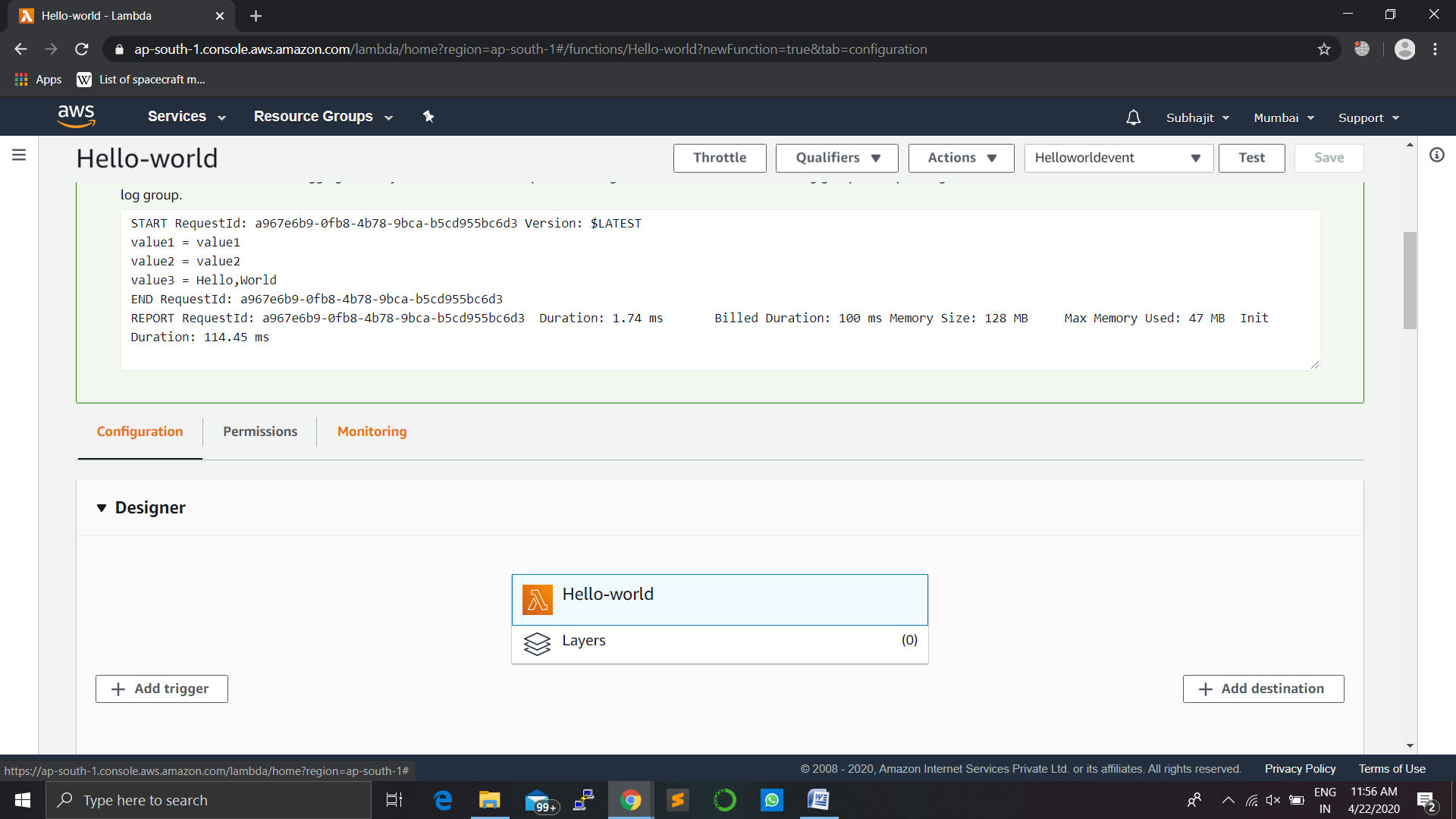


## Step 5: Monitor Your Metrics

AWS Lambda automatically monitors Lambda functions and reports metrics through Amazon CloudWatch. To help you monitor your code as it executes, Lambda automatically tracks the number of requests, the latency per request, and the number of requests resulting in an error and publishes the associated metrics.

a.  Invoke the Lambda function a few more times by repeatedly clicking the Test button.  This will generate the metrics that can be viewed in the next step.

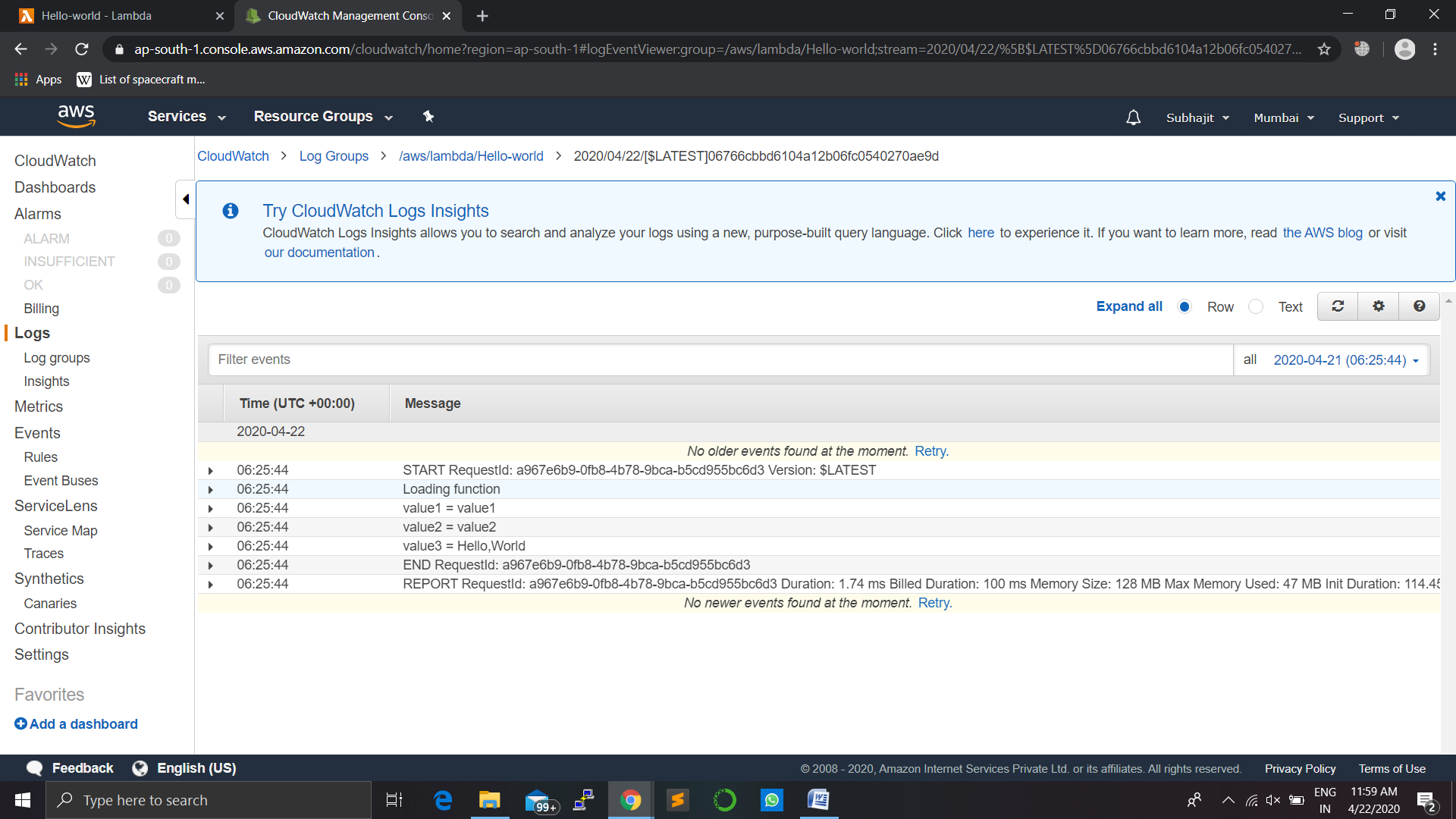
b.  Select Monitoring to view the results.



c.  Scroll down to view the metrics for your Lambda function.  Lambda metrics are reported through Amazon CloudWatch. You can leverage these metrics to set custom alarms. For more information about CloudWatch, see the [Amazon CloudWatch Developer Guide](http://docs.aws.amazon.com/AmazonCloudWatch/latest/DeveloperGuide/WhatIsCloudWatch.html).

The Monitoring tab will show six CloudWatch metrics:*Invocation count, Invocation duration, Invocation errors*, *Throttled invocations, Iterator age, and DLQ errors.*

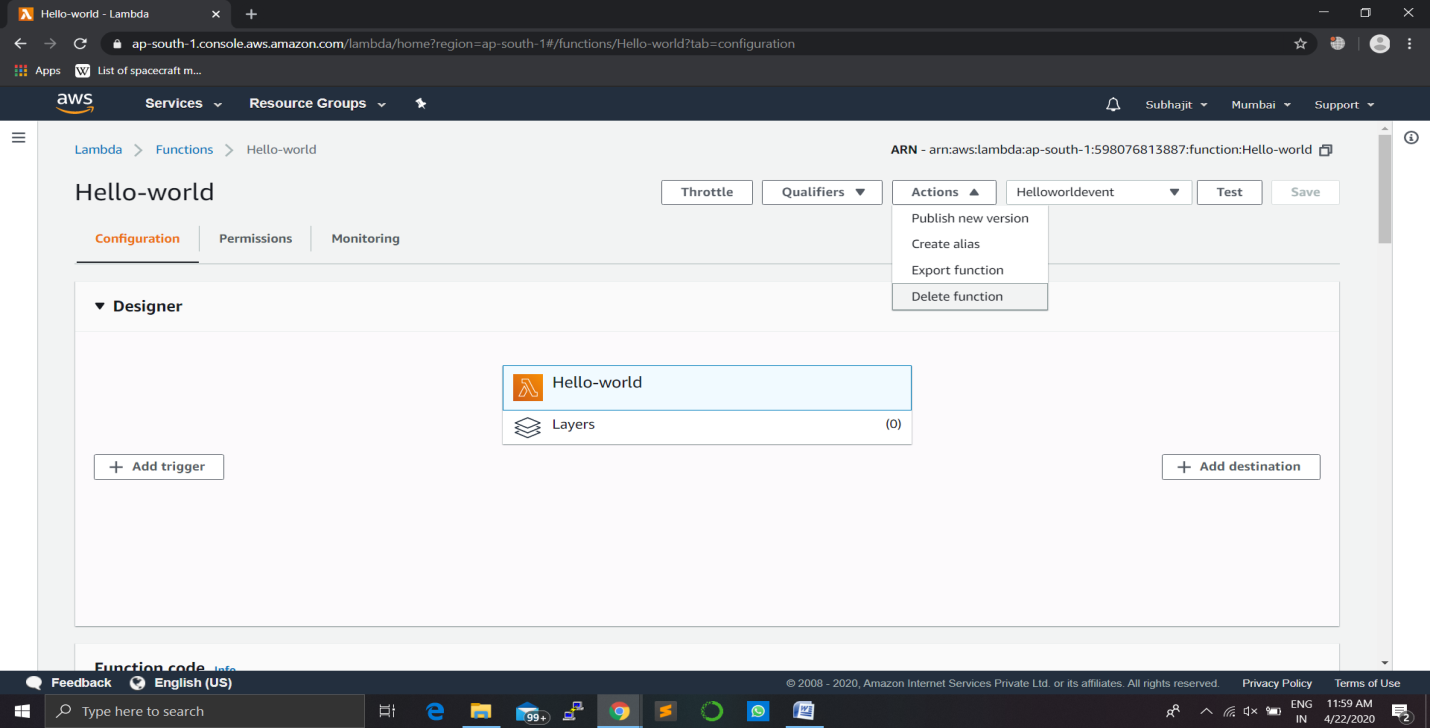
With AWS Lambda, you pay for what you use. After you hit your AWS Lambda free tier limit, you are charged based on the number of requests for your functions (invocation count) and the time your code executes (invocation duration).  For more information, see [AWS Lambda Pricing](https://aws.amazon.com/lambda/pricing/).



## Step 6: Delete the Lambda Function

While you will not get charged for keeping your Lambda function, you can easily delete it from the AWS Lambda console.

a.  Select the Actions button and click Delete Function.



b.  You will be asked to confirm your termination - select Delete.

