**EXPERIMENT NO. – 11**

**AIM:** To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

**LO:** LO1:- To understand the fundamentals of Cloud Computing and be fully proficient with Cloud based DevOps solution deployment options to meet your business requirements

LO6:- To engineer a composition of nano services using AWS Lambda and Step Functions with the Serverless Framework

**THEORY:**

AWS Lambda is a serverless, event-driven compute service that lets you run code for virtually any type of application or backend service without provisioning or managing servers. You can trigger Lambda from over 200 AWS services and software as a service (SaaS) applications, and only pay for what you use.

**AWS Lambda** is an [event-driven](https://en.wikipedia.org/wiki/Event-driven_programming), [serverless computing](https://en.wikipedia.org/wiki/Serverless_computing" \o "Serverless computing) platform provided by [Amazon](https://en.wikipedia.org/wiki/Amazon.com) as a part of [Amazon Web Services](https://en.wikipedia.org/wiki/Amazon_Web_Services). It is a computing service that runs code in response to [events](https://en.wikipedia.org/wiki/Event_(computing)) and automatically manages the computing resources required by that code. It was introduced on November 13, 2014.

[Node.js](https://en.wikipedia.org/wiki/Node.js), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [Go](https://en.wikipedia.org/wiki/Go_(programming_language)), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)" \o "Ruby (programming language)),and [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)) (through [.NET](https://en.wikipedia.org/wiki/.NET)) are all officially supported as of 2018. In late 2018, custom runtime support was added to AWS Lambda.

AWS Lambda supports running [native](https://en.wikipedia.org/wiki/Native_code) [Linux](https://en.wikipedia.org/wiki/Linux) [executables](https://en.wikipedia.org/wiki/Executable) via calling out from a supported runtime such as Node.js.For example, [Haskell](https://en.wikipedia.org/wiki/Haskell_(programming_language)) code can be run on Lambda.

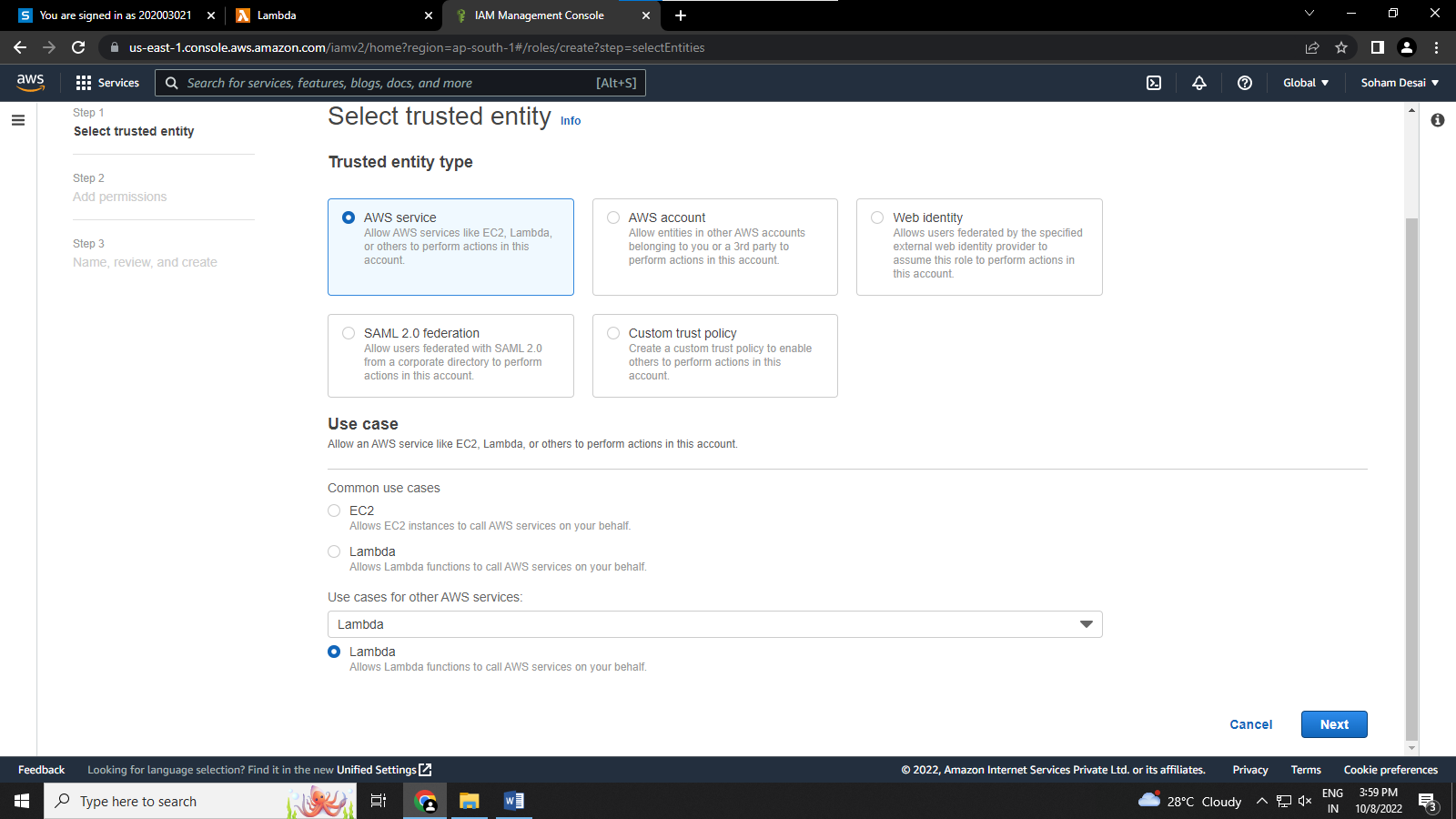
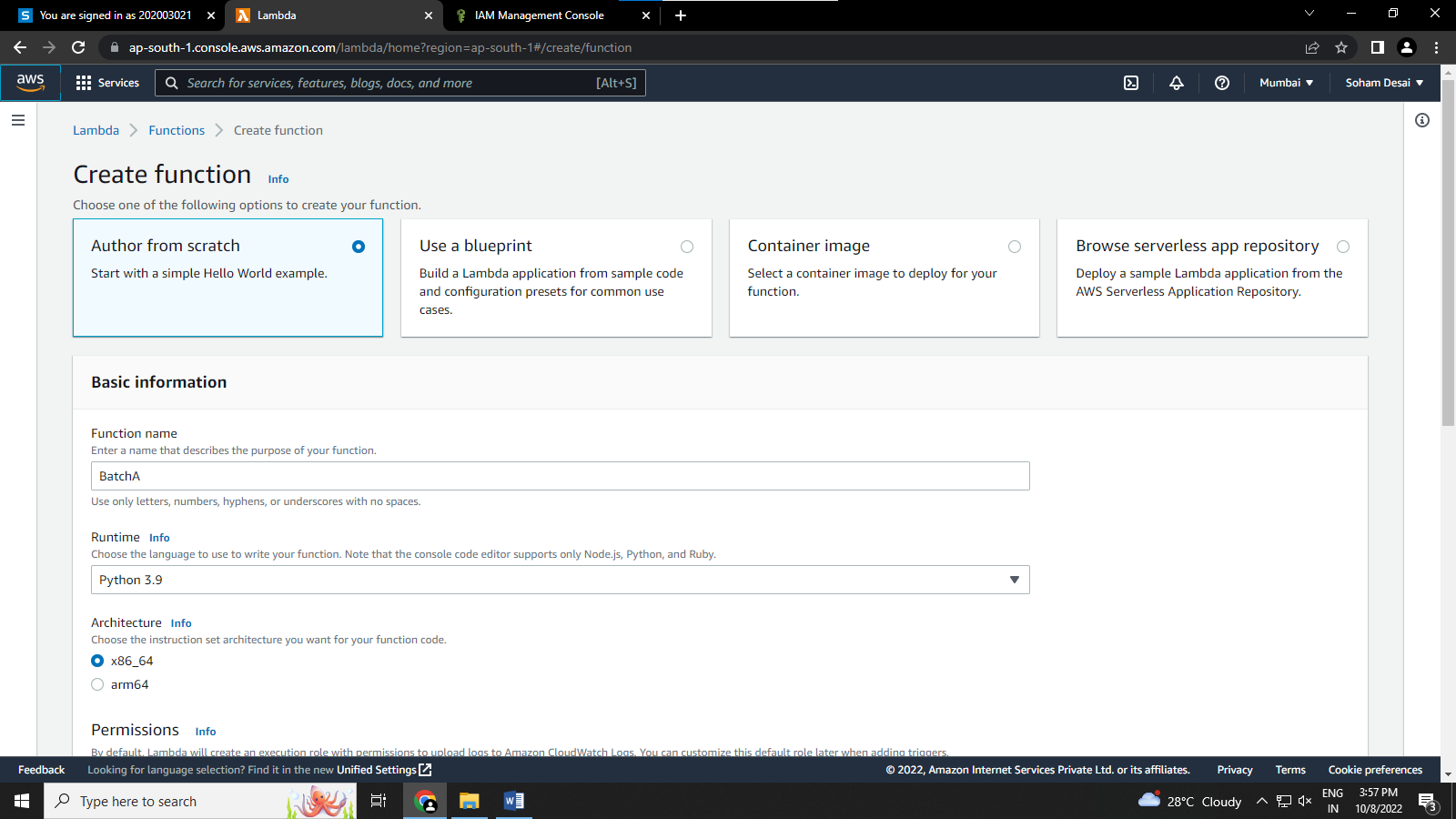
AWS Lambda was designed for use cases such as image or object [uploads](https://en.wikipedia.org/wiki/Upload) to [Amazon S3](https://en.wikipedia.org/wiki/Amazon_S3), updates to [DynamoDB](https://en.wikipedia.org/wiki/DynamoDB" \o "DynamoDB) tables, responding to website clicks, or reacting to sensor readings from an [IoT](https://en.wikipedia.org/wiki/Internet_of_Things" \o "Internet of Things) connected device. AWS Lambda can also be used to automatically provision back-end services triggered by custom [HTTP requests](https://en.wikipedia.org/wiki/HTTP_request), and "spin down" such services when not in use, to save resources. These custom HTTP requests are configured in AWS API Gateway, which can also handle [authentication](https://en.wikipedia.org/wiki/Authentication) and [authorization](https://en.wikipedia.org/wiki/Authorization) in conjunction with [AWS Cognito](https://en.wikipedia.org/wiki/Amazon_Cognito).

Unlike Amazon EC2, which is priced by the hour but metered by the second, AWS Lambda is metered by rounding up to the nearest millisecond with no minimum execution time.

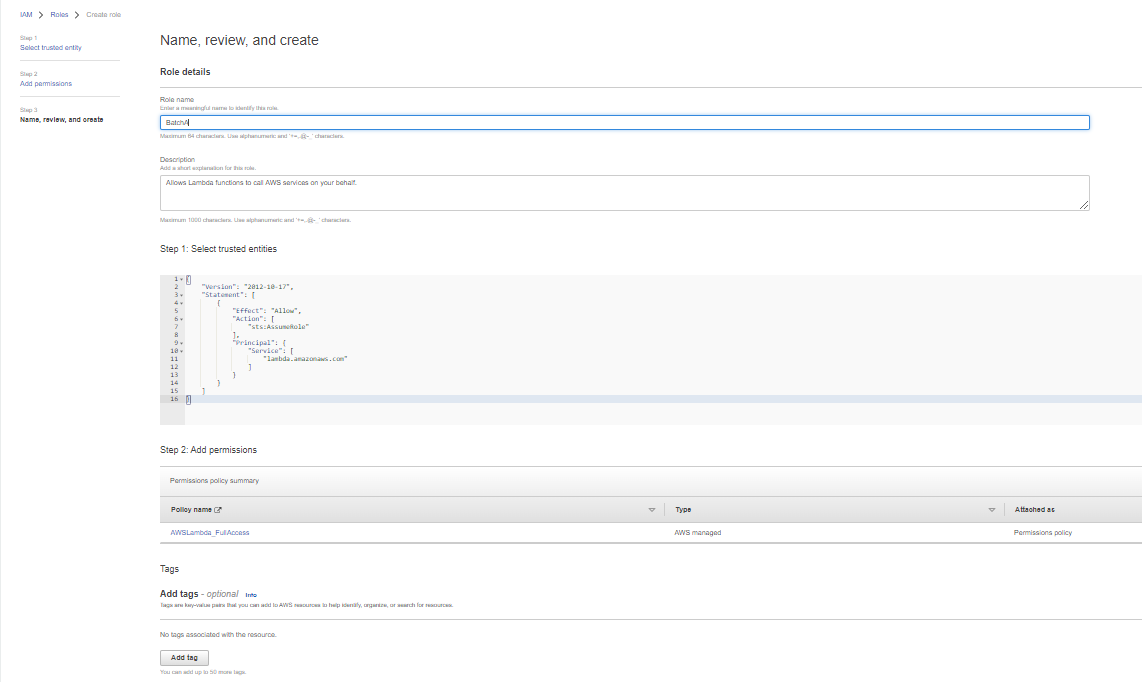
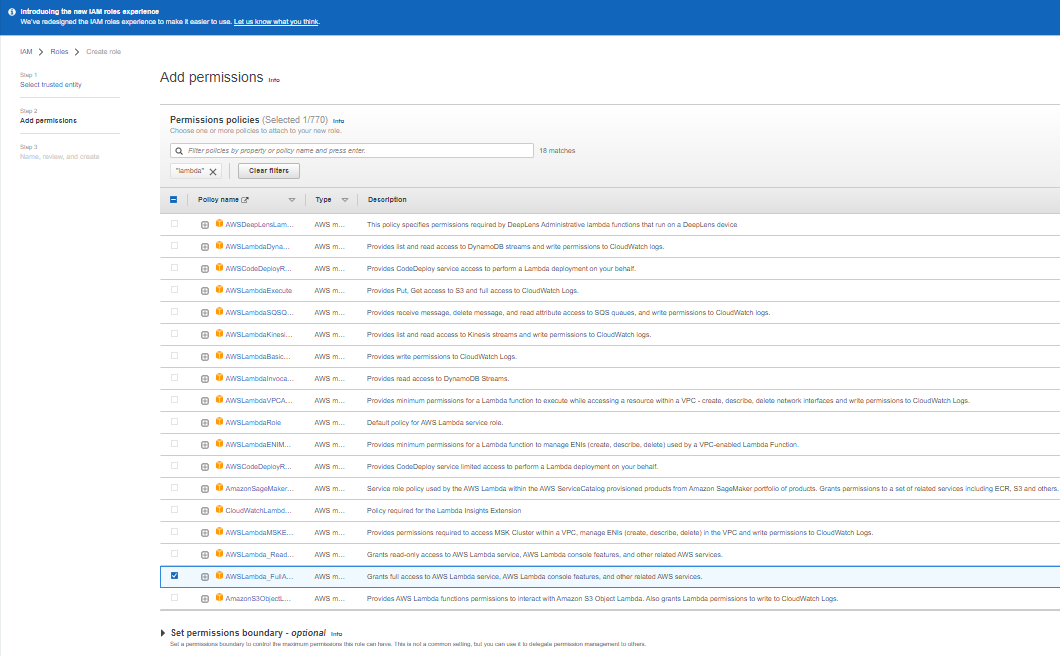
In 2019, at AWS' annual cloud computing conference (AWS re:Invent), the AWS Lambda team announced "Provisioned Concurrency", a feature that "keeps functions initialized and hyper-ready to respond in double-digit milliseconds." The Lambda team described Provisioned Concurrency as "ideal for implementing interactive services, such as web and mobile backends, latency-sensitive microservices, or synchronous APIs."

**Output:-**

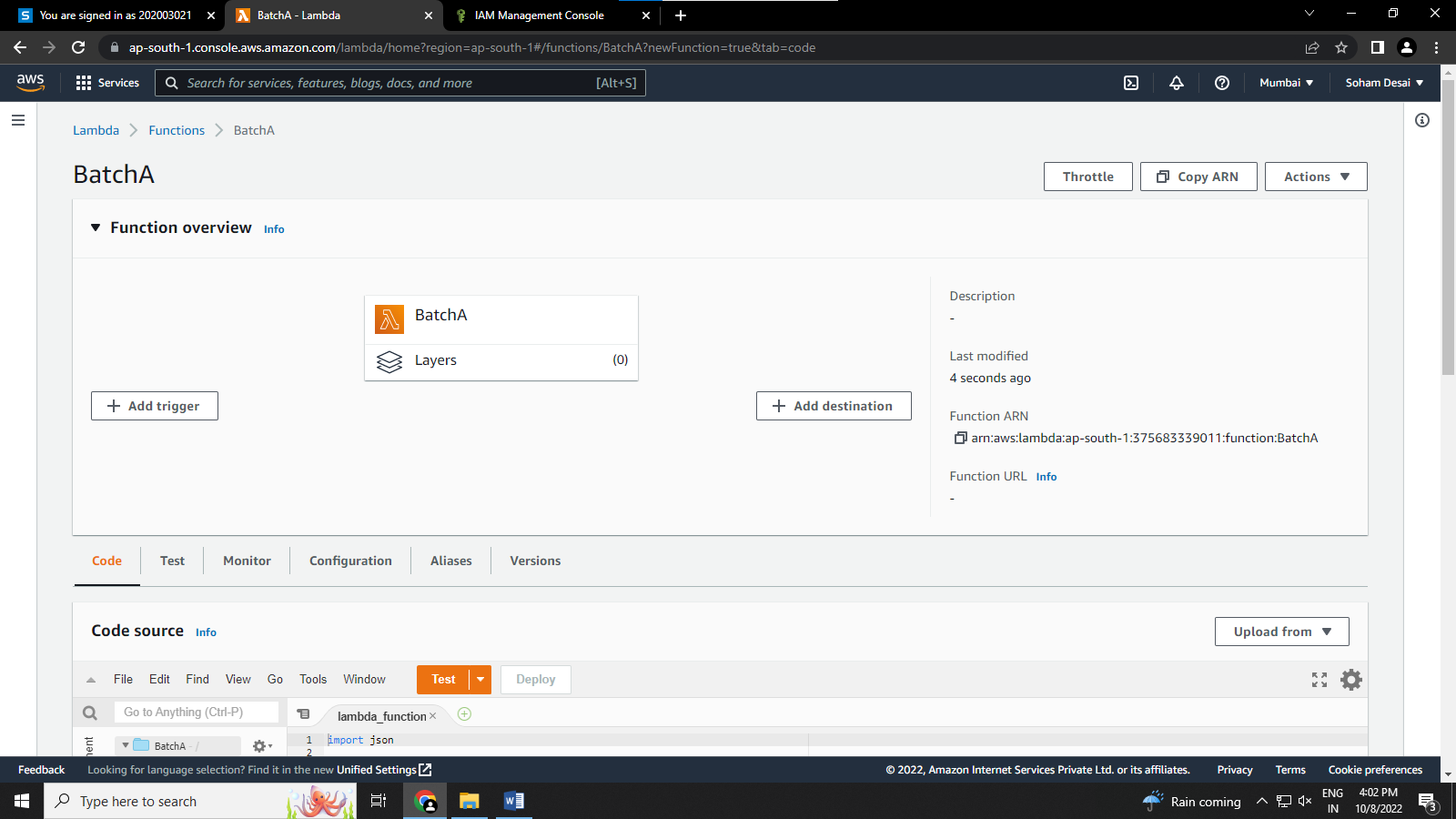
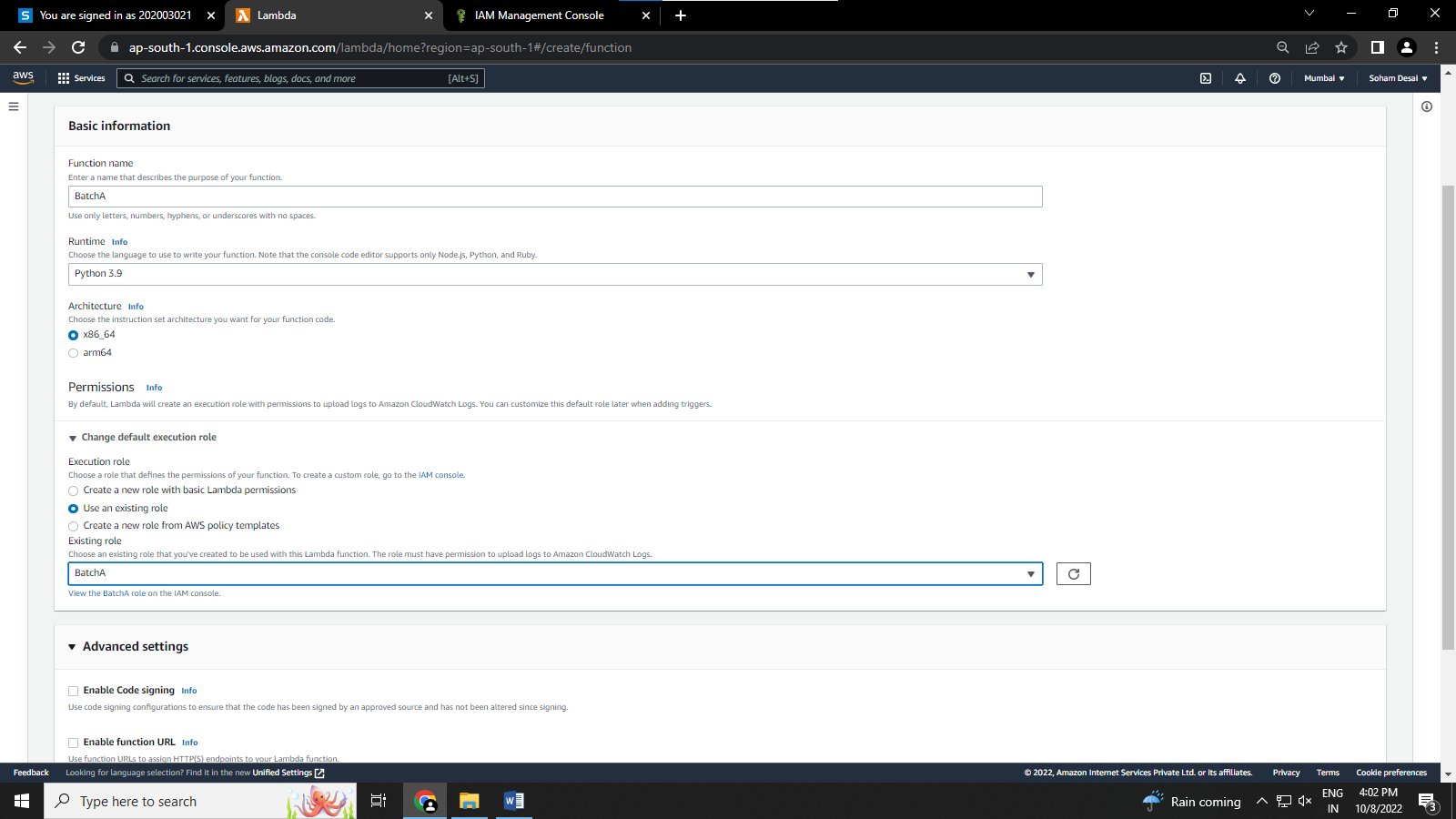
1. Create a lambda function



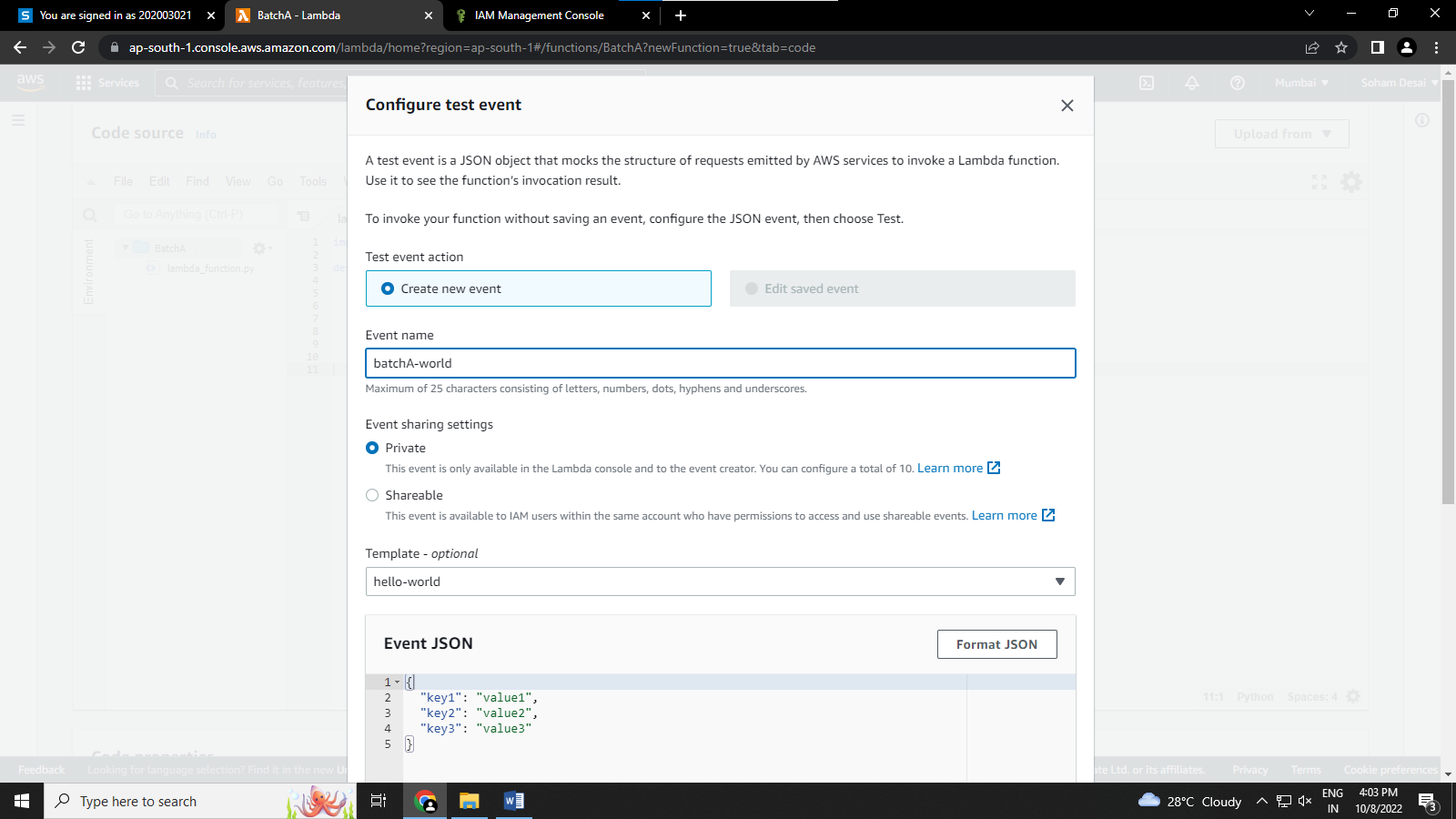
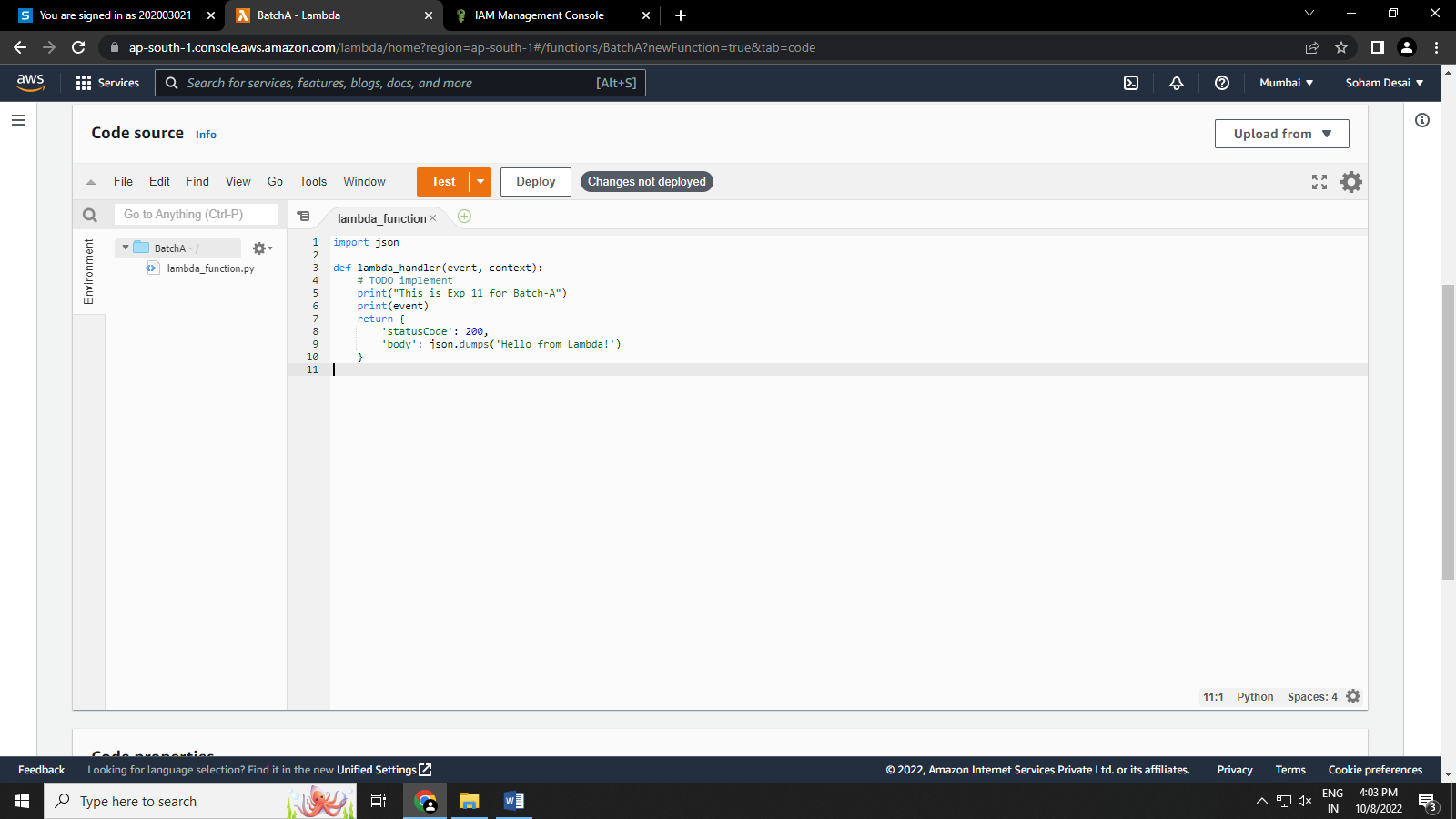
2) select the required policies



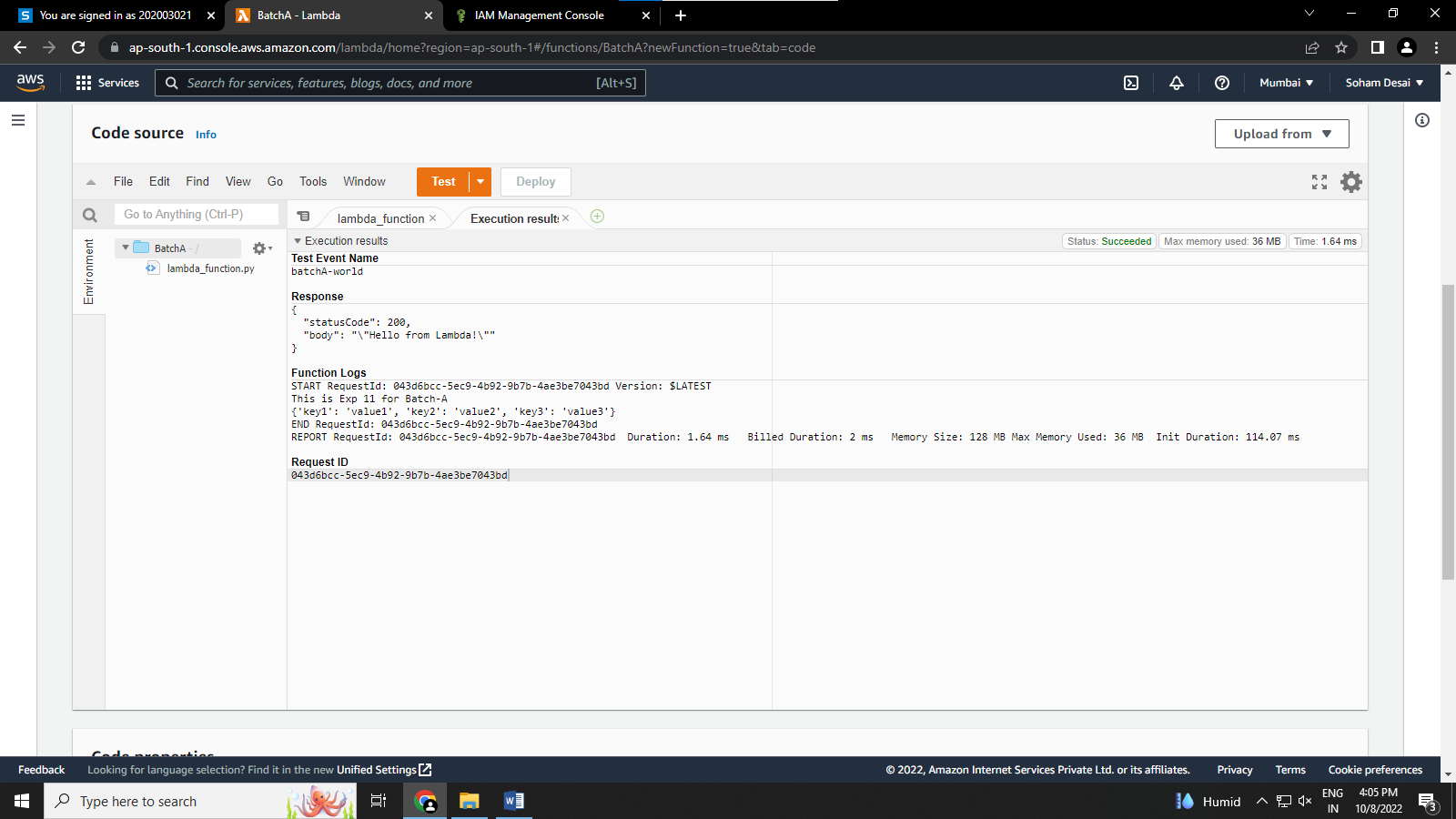
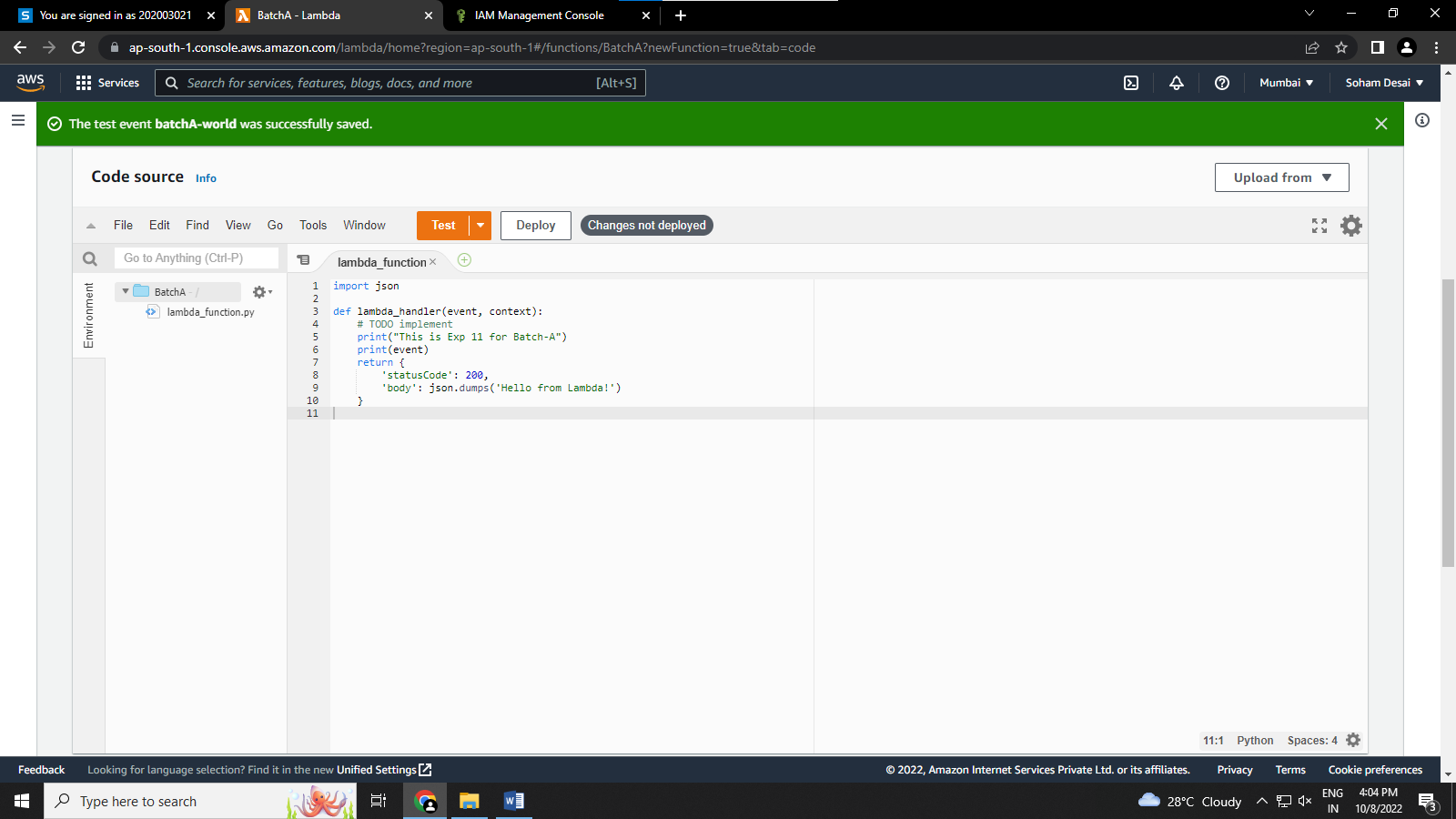
3) enter the required information like the architecture and language and create the function



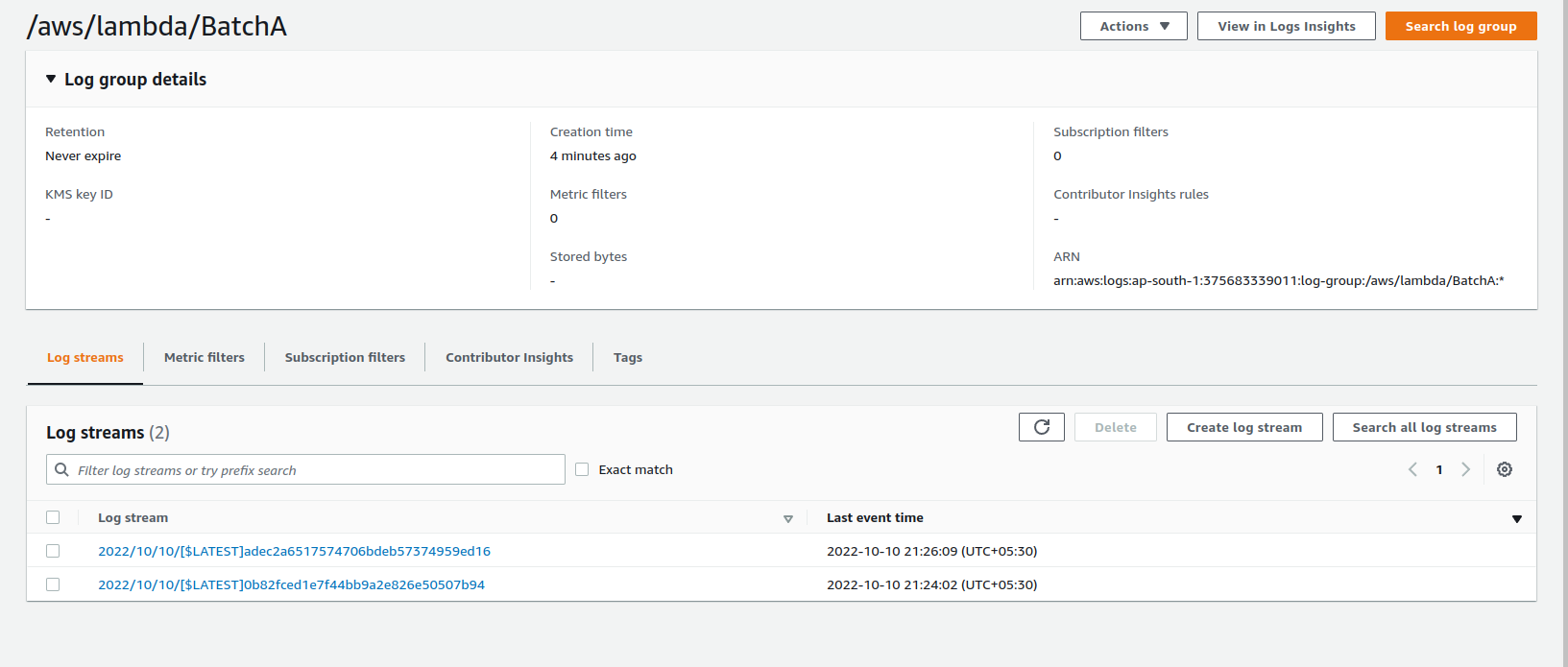
4) Write the test code for the function and configure the settings



5) Save the test and run it to check if it is working properly



6) check the log stream to see that the lambda function is present or not



**Conclusion:-**