Name: Manav Jawrani

Roll No.: 19

Subject: Advanced DevOps

Experiment No.: 11

Experiment 11

Aim: To understand AWS Lambda, it's workflow, various functions and create your first Lamda functions using Python / Java / Node.js.

Theory:

- What is AWS Lambda?

AWS Lamda is a severless computing service provided by Amazon web services. Users of AWS Lambda create functions, self-contained applications written in one of the supported languages and runtimes and upload them to AWS Lambda, which executes those functions in an efficient and flexible manner.
The Lambda function can perform any kind of computing task, from serving web pages and processing streams of data to calling APIs and integrating with other AWS services.
The concept of "serverless" computing refers to not needing to maintain your own severs to run these functions. AWS Lambda is fully managed service that takes care of all the infrastructure for you. And so "severless" doesn't mean that there are no servers involved, it just means that the severs, the operating systems, the network layer and the rest of infrastructure have already been taken care of so that you can focus on writing application Code.

- Components of Aws Lambda Function.
The three components of AWS Lambda are:

a. A function - This is the actual code that performs a task.
b. A configuration - This specifies how your function is executed.
c. An event source - This is the event that triggers the function. You can trigger with several AWS services or a third party service.

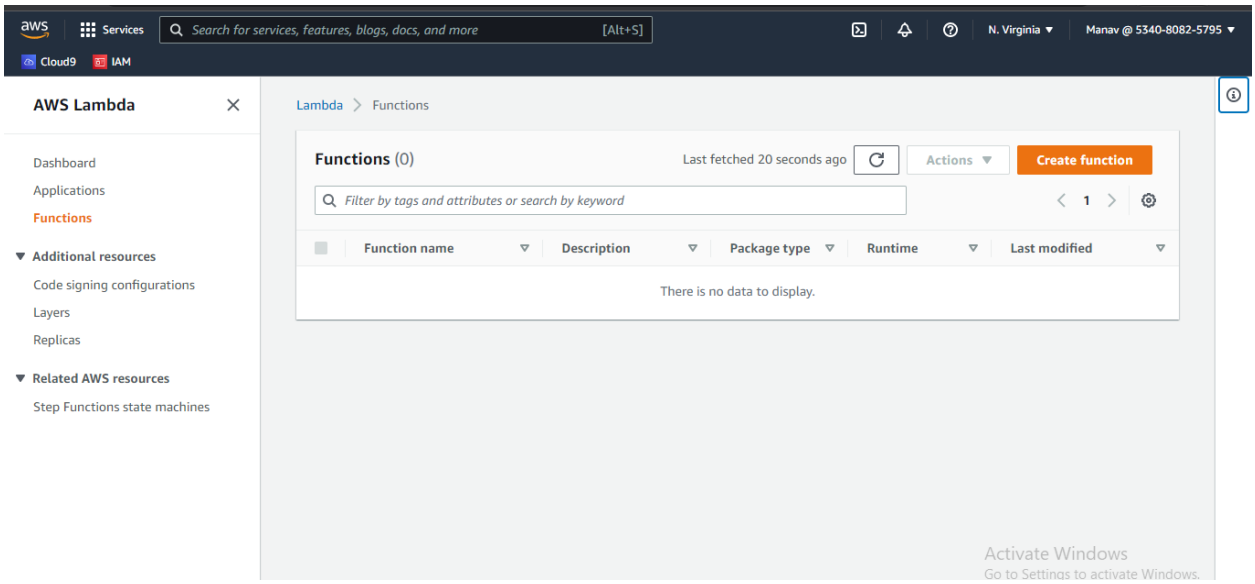
- Features of AWS Lambda.
1. It scales the infrastructure without any additional configuration and reduces the operational work.
2. It offers multiple options like AWS S3, CloudWatch, Code commit and many more to trigger an event.
3. AWS Lambda is secure, it ufes IAM to define all the roles and security policies.

Implementation:

Prerequisites:

1. An AWS Account

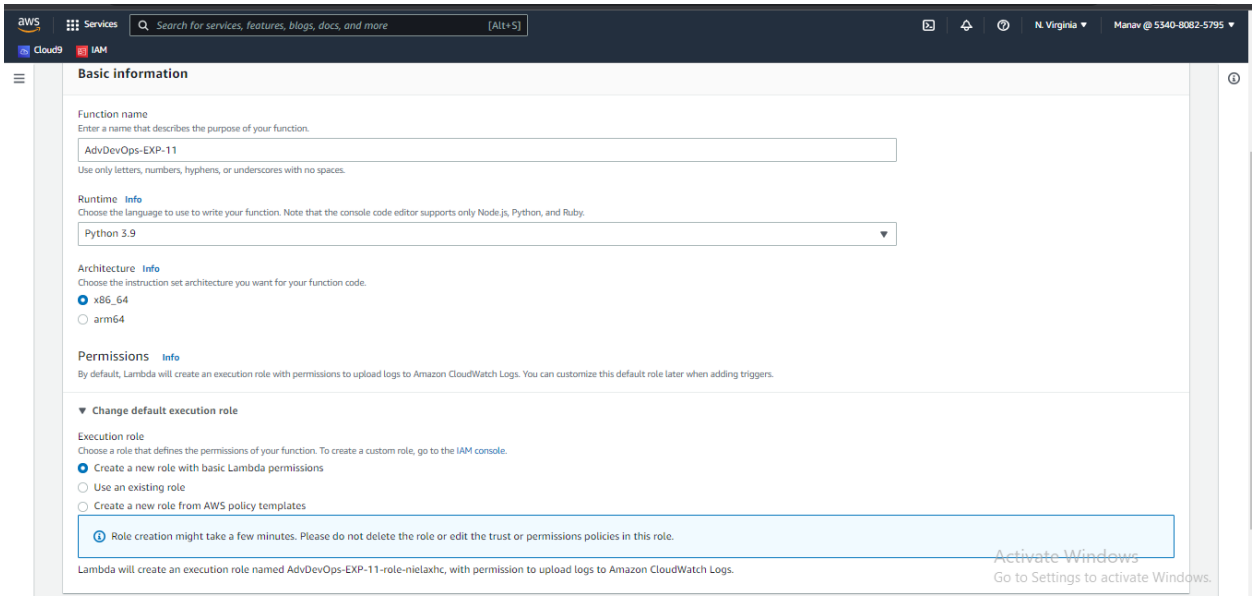
Step 1: Open up the Lambda Console and click on the Create button. Be mindful of where you create your functions since Lambda is region-dependent.



Step 2: Choose to create a function from scratch or use a blueprint, i.e templates defined by AWS for you with all configuration presets required for the most common use cases.

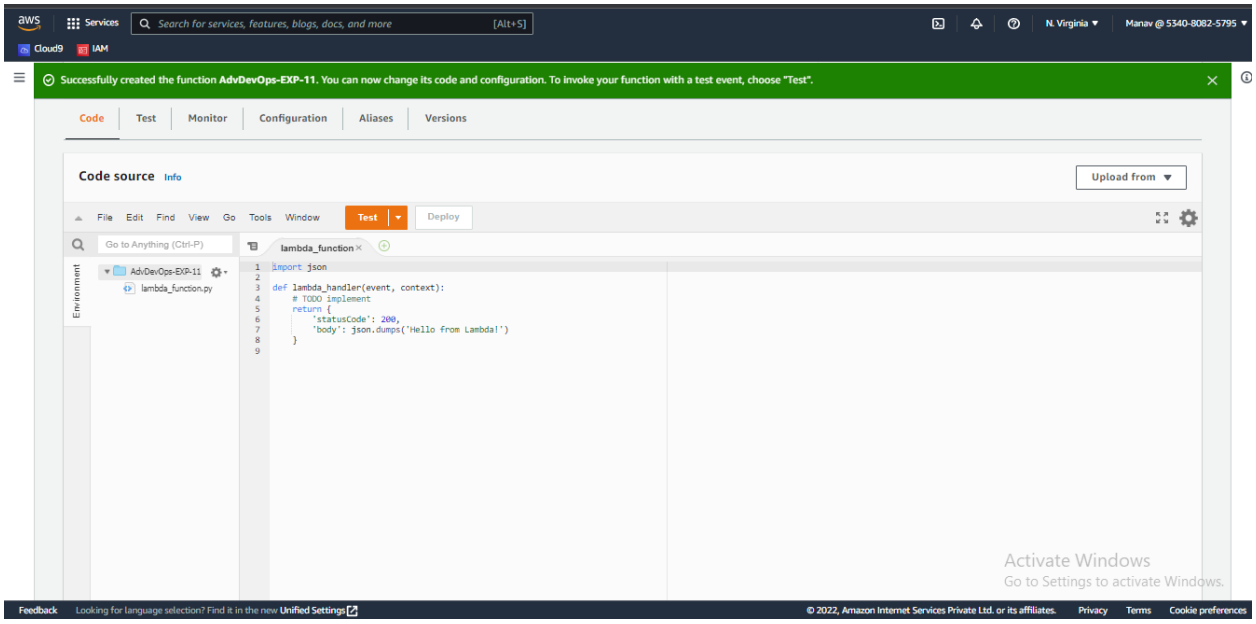
Then, choose a runtime env for your function, under the dropdown, you can see all the options AWS supports, Python, Nodejs, .NET and Java being the most popular ones.

After that, choose to create a new role with basic Lambda permissions if you don't have an existing one.

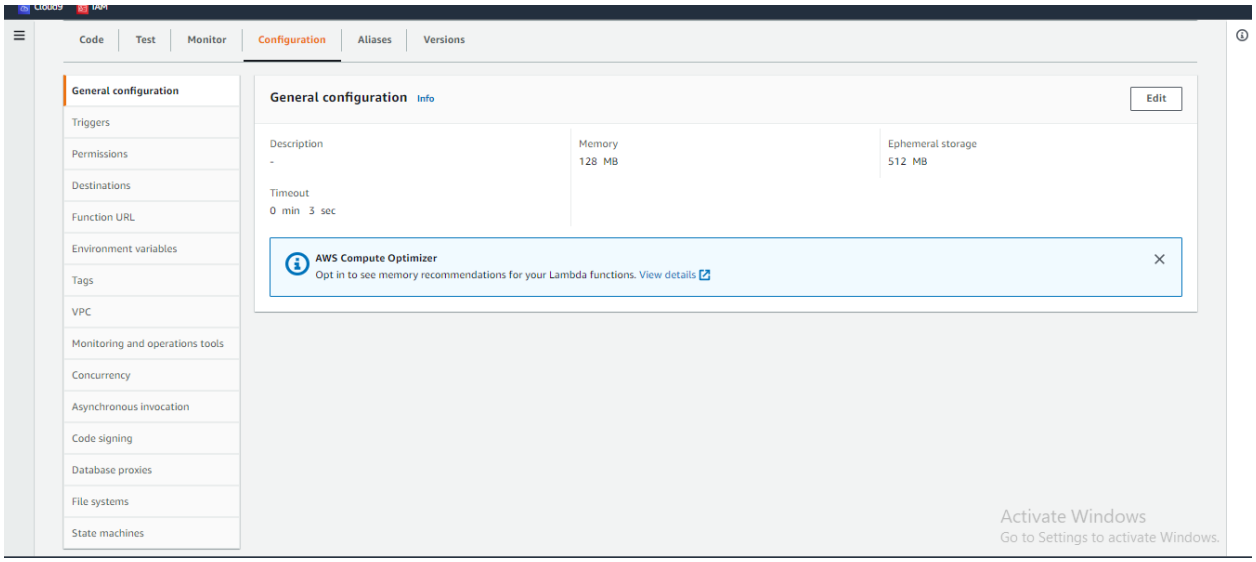


Click on the Create button.

Step 3: This process will take a while to finish and after that, you'll get a message that your function was successfully created.



Step 4: To change the configuration, open up the Configuration Tab and under General Configuration, choose Edit. Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.



Basic settings Info

Description - *optional*

Memory Info

Your function is allocated CPU proportional to the memory configured.

128 MB

Set memory to between 128 MB and 10240 MB

Ephemeral storage Info

You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing

512 MB

Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

Timeout

0 min 1 sec

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

- (•) Use an existing role
- ( ) Create a new role from AWS policy templates

Existing role

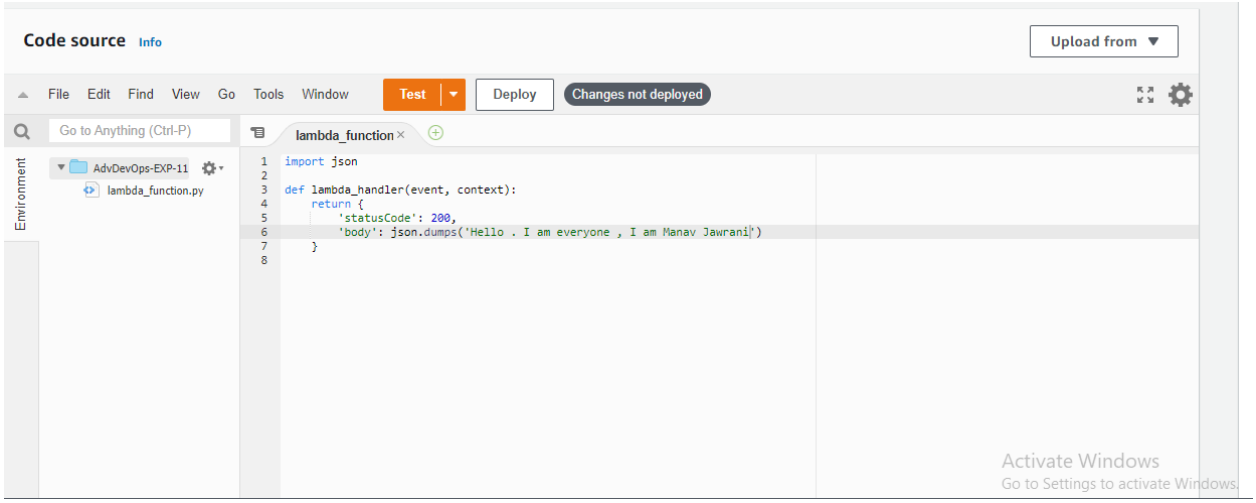
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

service-role/AdvDevOps-EXP-11-role-nielaxhc

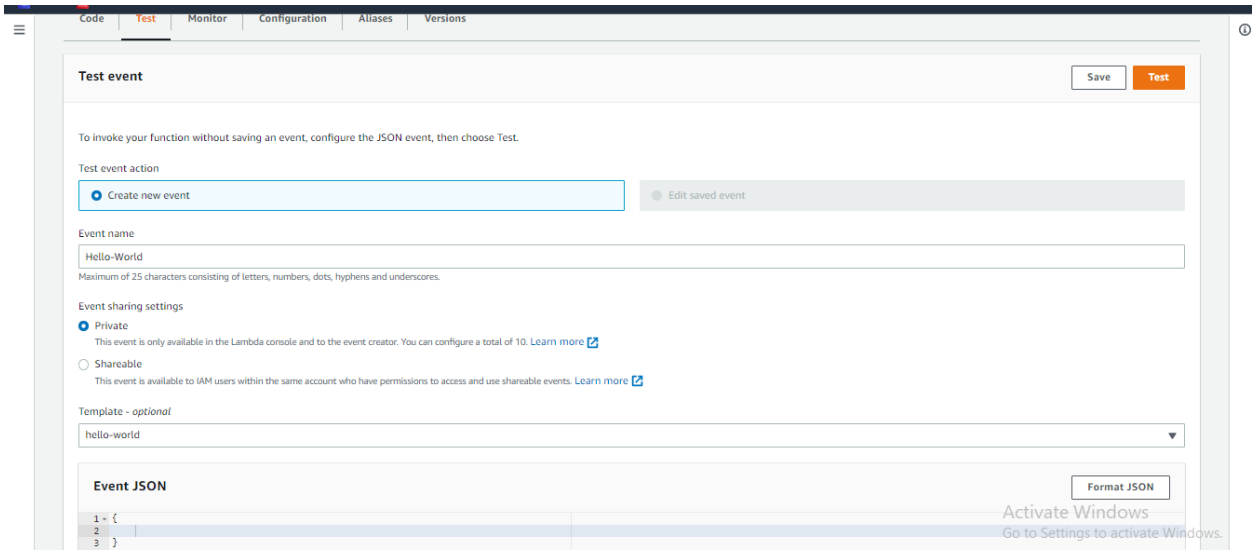
View the AdvDevOps-EXP-11-role-nielaxhc role on the IAM console.

Cancel Save

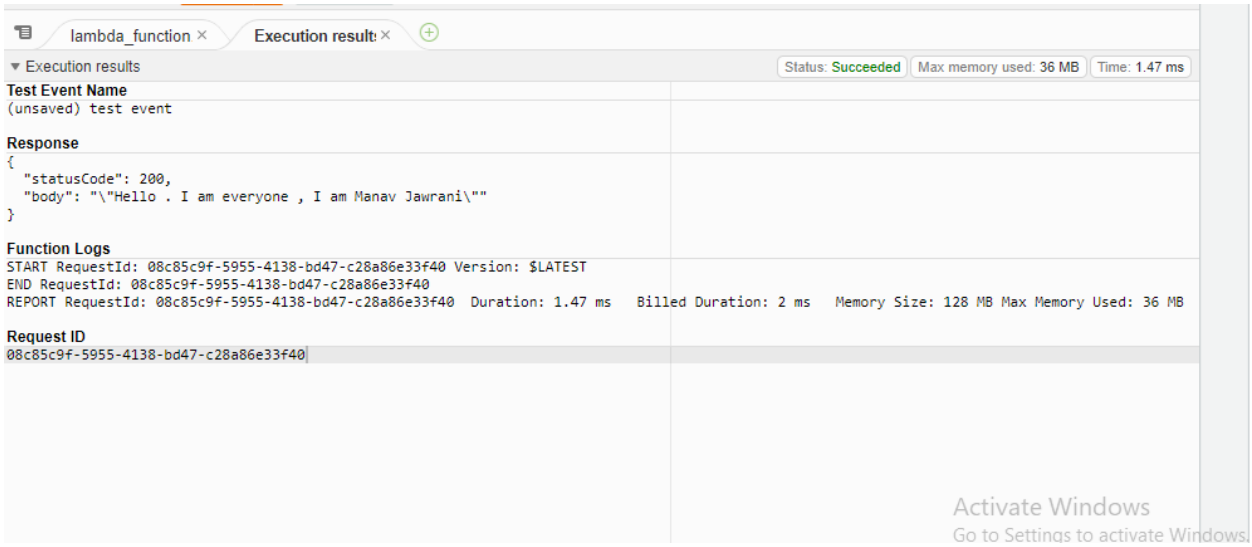
Step 5: You can make changes to your function inside the code editor. You can also upload a zip file of your function or upload one from an S3 bucket if needed. Press Ctrl + S to save the file and click Deploy to deploy the changes.



Step 6: Click on Test and you can change the configuration, like so. If you do not have anything in the request body, it is important to specify two curly braces as valid JSON, so make sure they are there.



Step 7: Now click on Test and you should be able to see the results.



Conclusion:

In this experiment, we learnt about AWS Lambda function, its features and how the serverless system can be designed and how it function. Also, we used it to create, deploy and test serverless functions in the cloud.