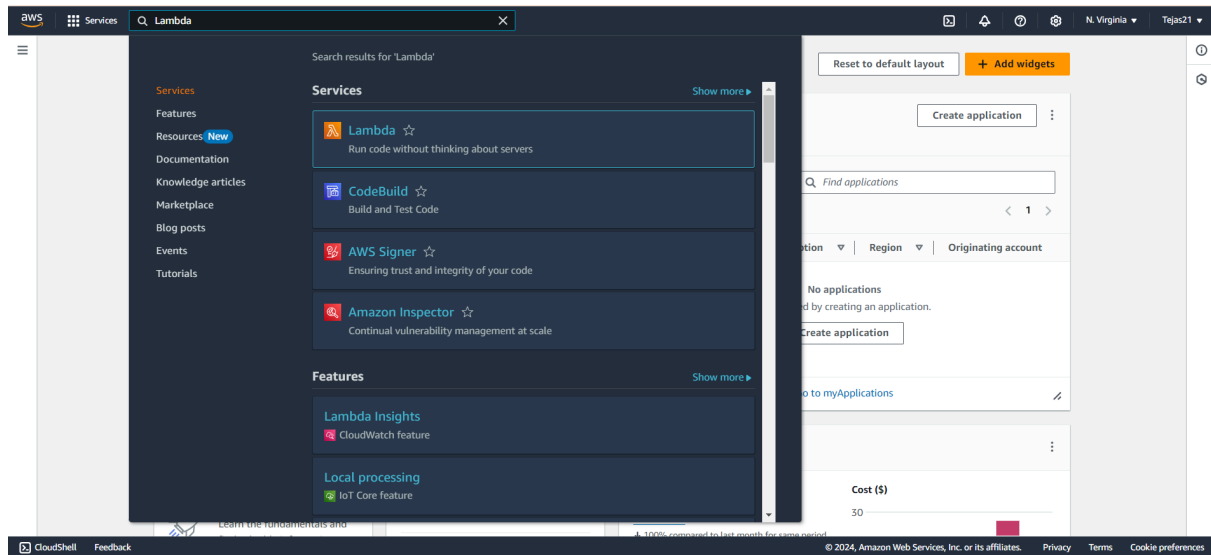


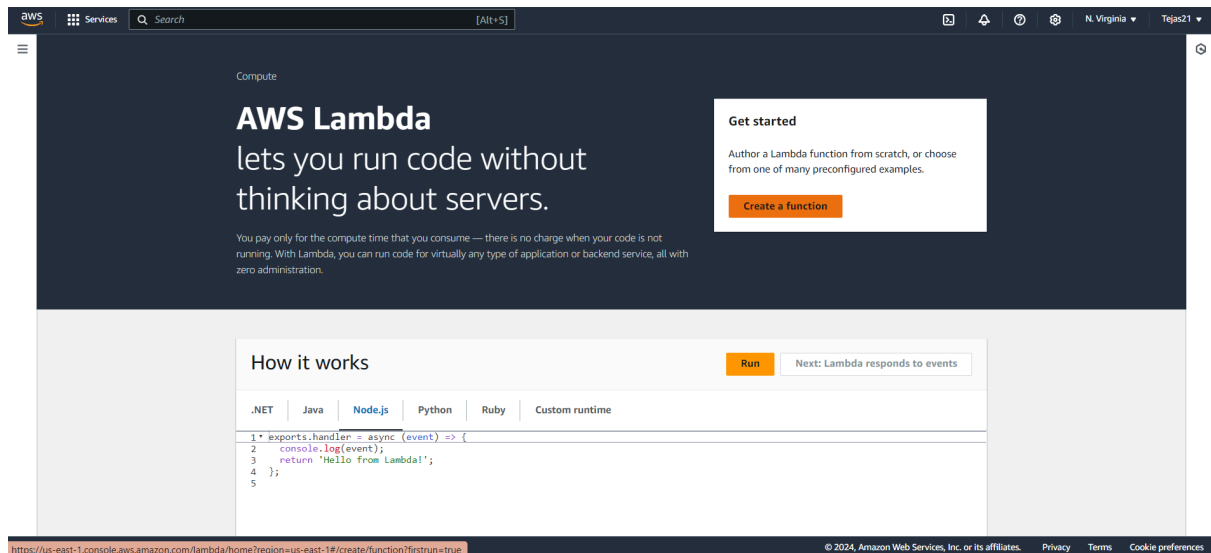
EXPERIMENT NO: - 11

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

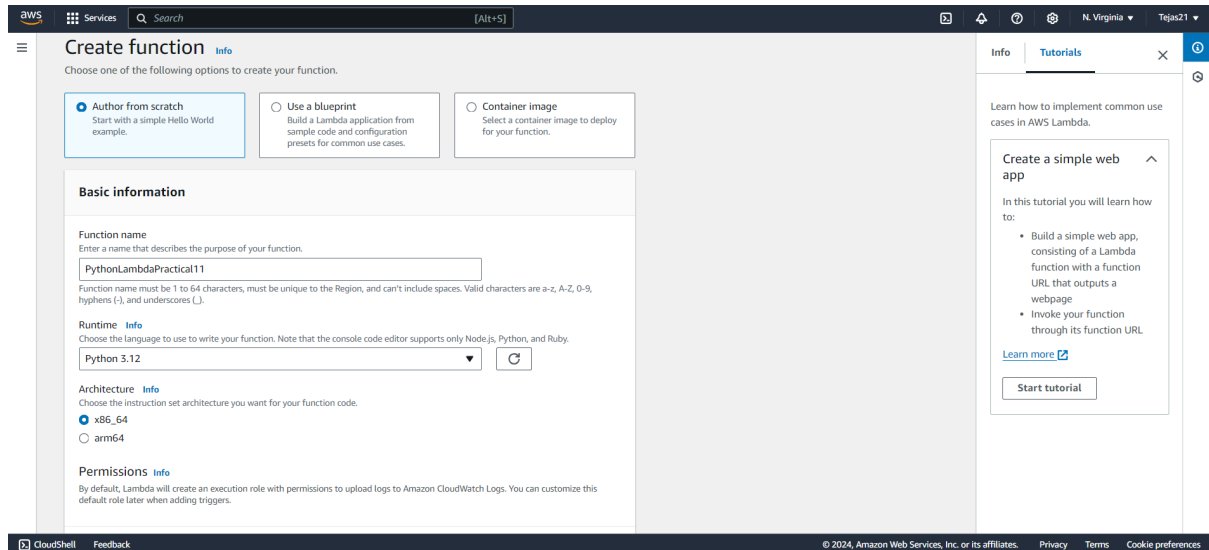
- ❑ Open Lambda to create an AWS Lambda function



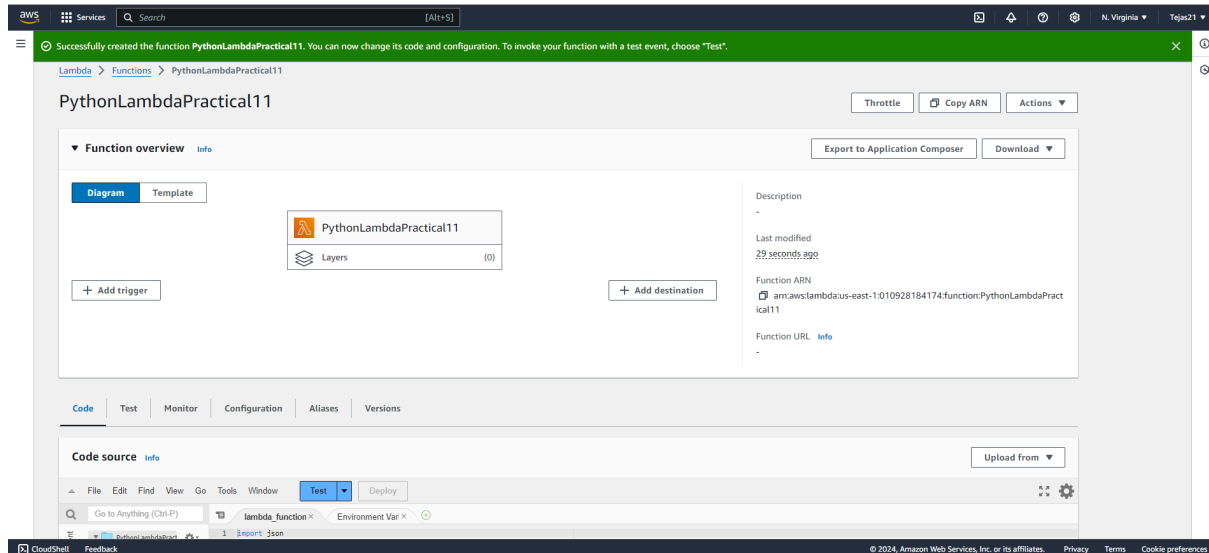
- ❑ Open up the Lambda Console and click on the Create button.

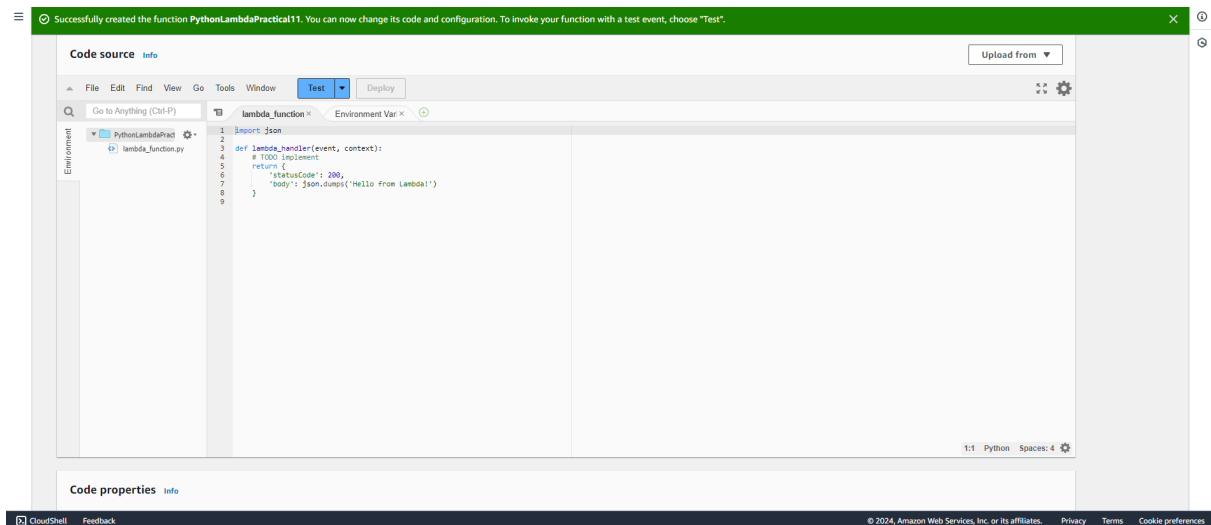


- ❑ Choose to create a function from scratch or use a blueprint, i.e templates defined by AWS for you with all configuration presets required for the most common use cases. Then, choose a runtime env for your function, under the dropdown, you can see all the options AWS supports, Python, Nodejs, .NET 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.

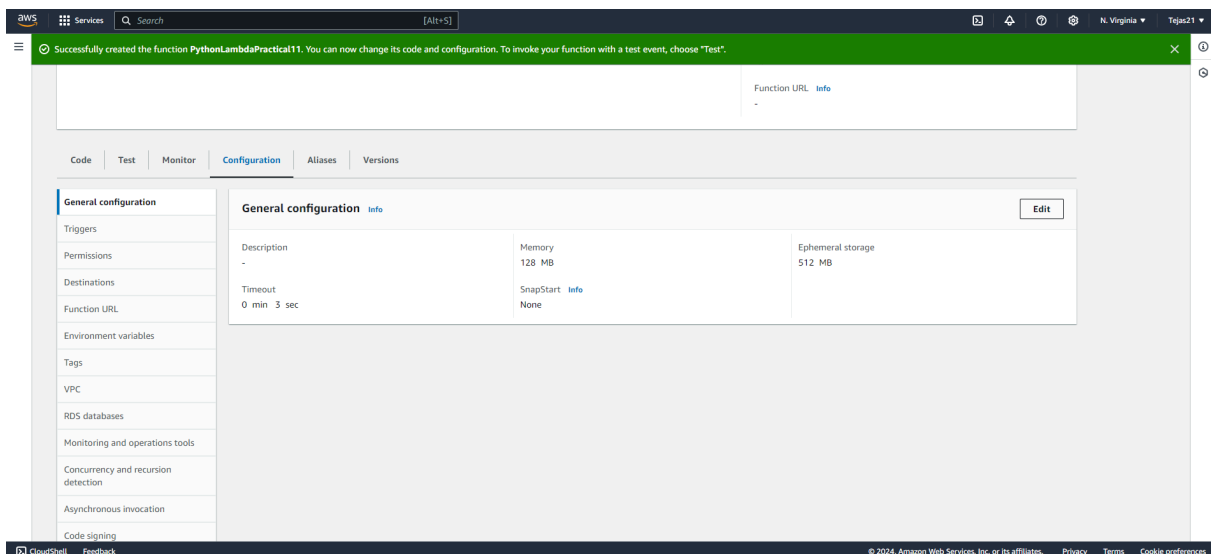


- ❑ Function is successfully created





- ☐ 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.



The screenshot shows the AWS Lambda console configuration page for a new function. The page is titled "Description - optional" and includes several configuration sections:

- Memory:** A dropdown menu set to "128 MB". Below it, text states: "Your function is allocated CPU proportional to the memory configured. Set memory to between 128 MB and 10240 MB."
- Ephemeral storage:** A dropdown menu set to "512 MB". Below it, text states: "You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing". Below that, text states: "Set ephemeral storage (/tmp) to between 512 MB and 10240 MB."
- SnapStart:** A dropdown menu set to "None". Below it, text states: "Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the SnapStart compatibility considerations".
- Timeout:** Two input fields: "0 min" and "1 sec".
- Execution role:** Two radio buttons: "Use an existing role" (selected) and "Create a new role from AWS policy templates".
- Existing role:** A dropdown menu showing "service-role/PythonLambdaPractical11-role-qwxs7w45". Below it, text states: "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." and "View the PythonLambdaPractical11-role-qwxs7w45 role on the IAM console."

At the bottom of the configuration section, there are "Cancel" and "Save" buttons. The footer of the console shows "© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences".

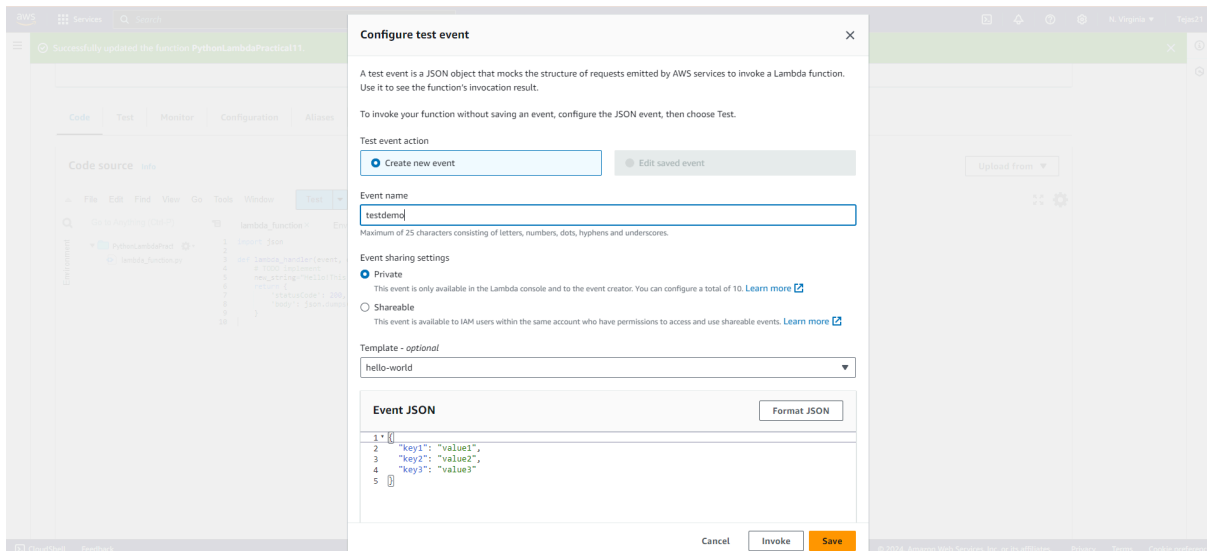
- ☐ You can make changes to your function inside the code editor. You can also upload a zip file of your function or upload one from an S3 bucket if needed. Press Ctrl + S to save the file and click Deploy to deploy the changes.

The screenshot shows the AWS Lambda console code editor for the function "PythonLambdaPractical11". A green banner at the top indicates "Successfully updated the function PythonLambdaPractical11." The editor has tabs for "Code", "Test", "Monitor", "Configuration", "Aliases", and "Versions". The "Code" tab is active, showing a Python function named "lambda_handler". The code is as follows:

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO: implement
5     new_string="Hello!This is lambda practical no 11"
6     return {
7         'statusCode': 200,
8         'body': json.dumps('Hello from Lambda!')}
9
10
```

The editor includes a file explorer on the left showing the file "lambda_function.py". At the top right of the editor, there are buttons for "Test" and "Deploy", and a status indicator "Changes not deployed". The footer of the console shows "© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences".

- ☐ 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.



- Click on Test and you should be able to see the results

