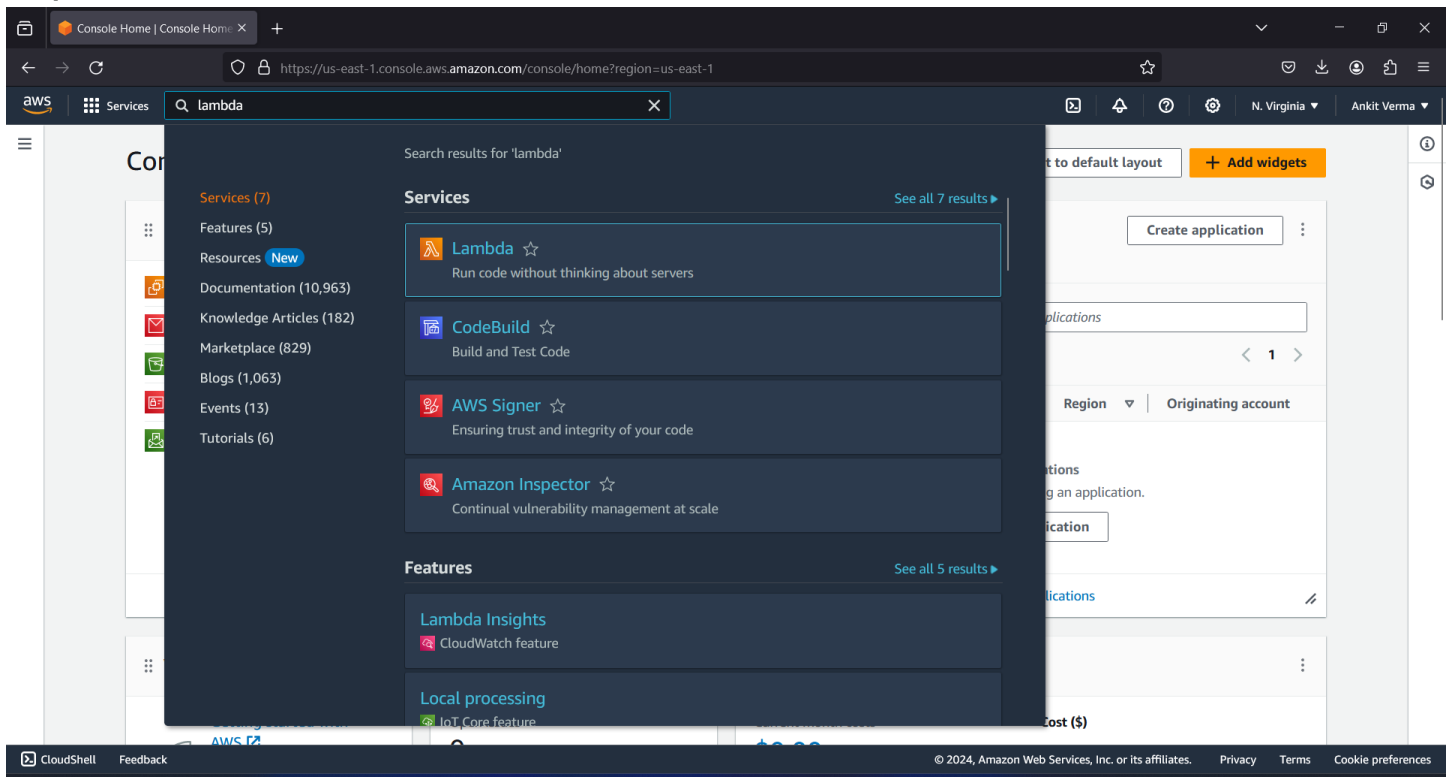


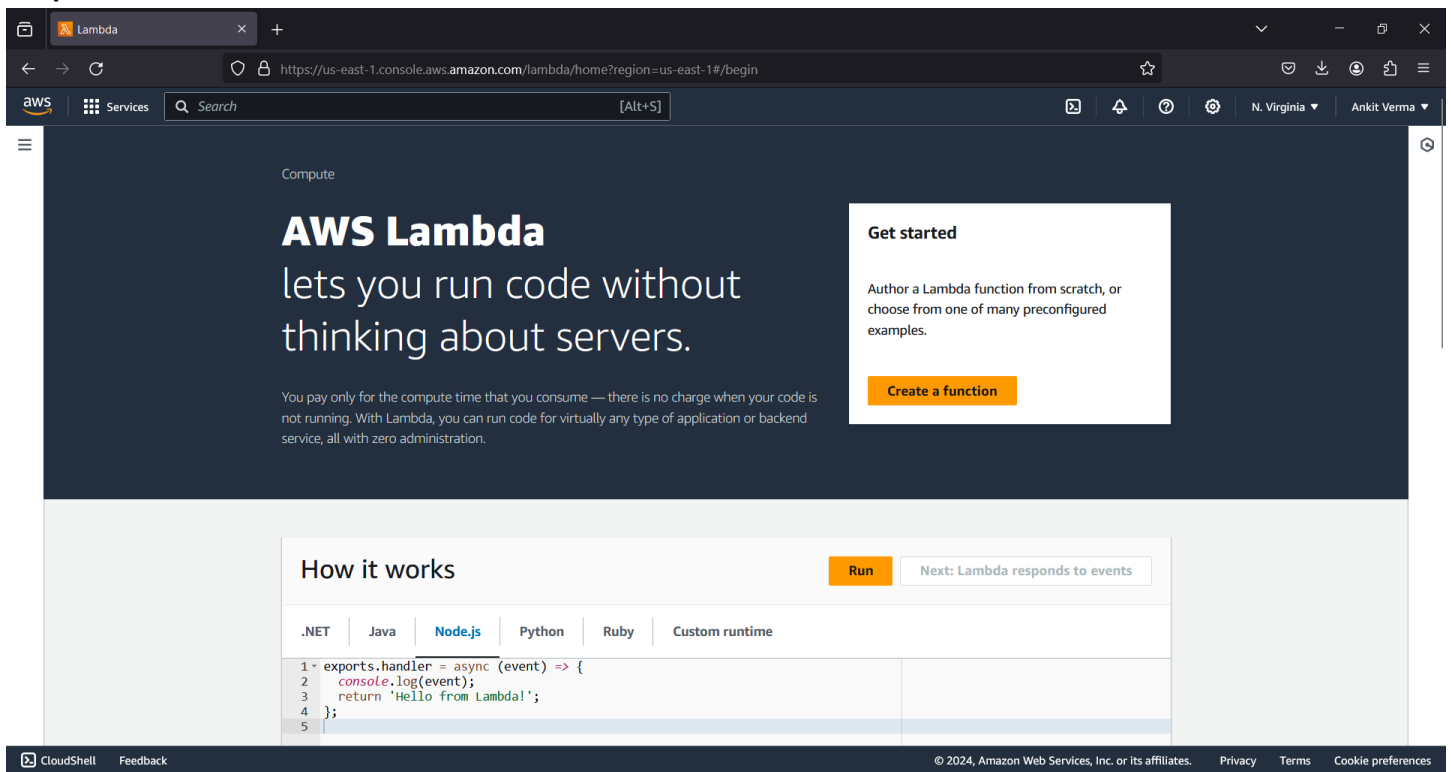
# Assignment: 15

Problem Statement: Create a Serverless Computing Service.

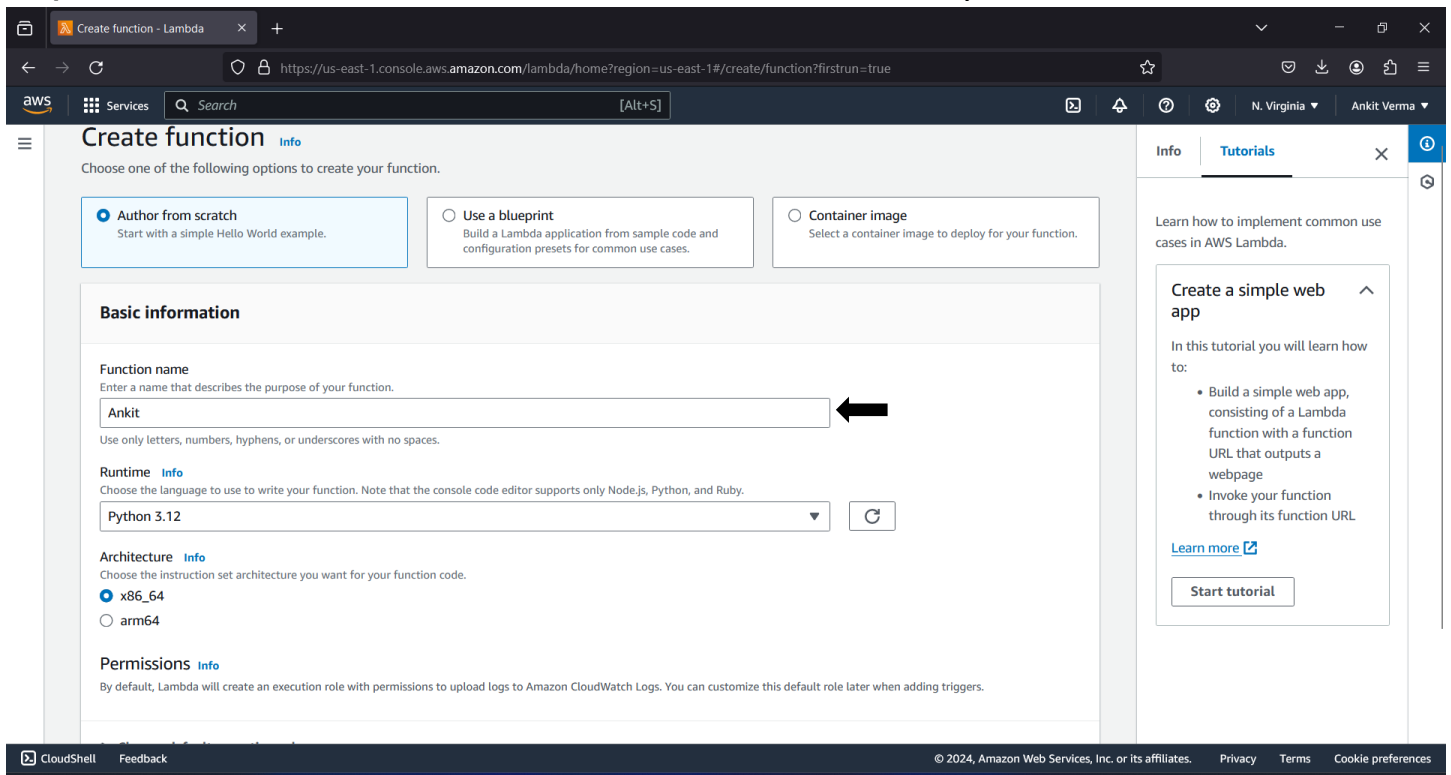
Step 1: Search for Lambda and click on it.



Step 2: Click on Create a function button.



### Step 3: Give a name to the function and under runtime select Python.




Create function - Lambda


Choose one of the following options to create your function.

- ☒ Author from scratch  
Start with a simple Hello World example.
- ☐ Use a blueprint  
Build a Lambda application from sample code and configuration presets for common use cases.
- ☐ Container image  
Select a container image to deploy for your function.

### Basic information

**Function name**  
Enter a name that describes the purpose of your function.  
 

Use only letters, numbers, hyphens, or underscores with no spaces.

**Runtime** [Info](#)  
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.  
 

**Architecture** [Info](#)  
Choose the instruction set architecture you want for your function code.  
☒ x86\_64  
☐ arm64

**Permissions** [Info](#)  
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

Info

Tutorials

Learn how to implement common use cases in AWS Lambda.

### Create a simple web app

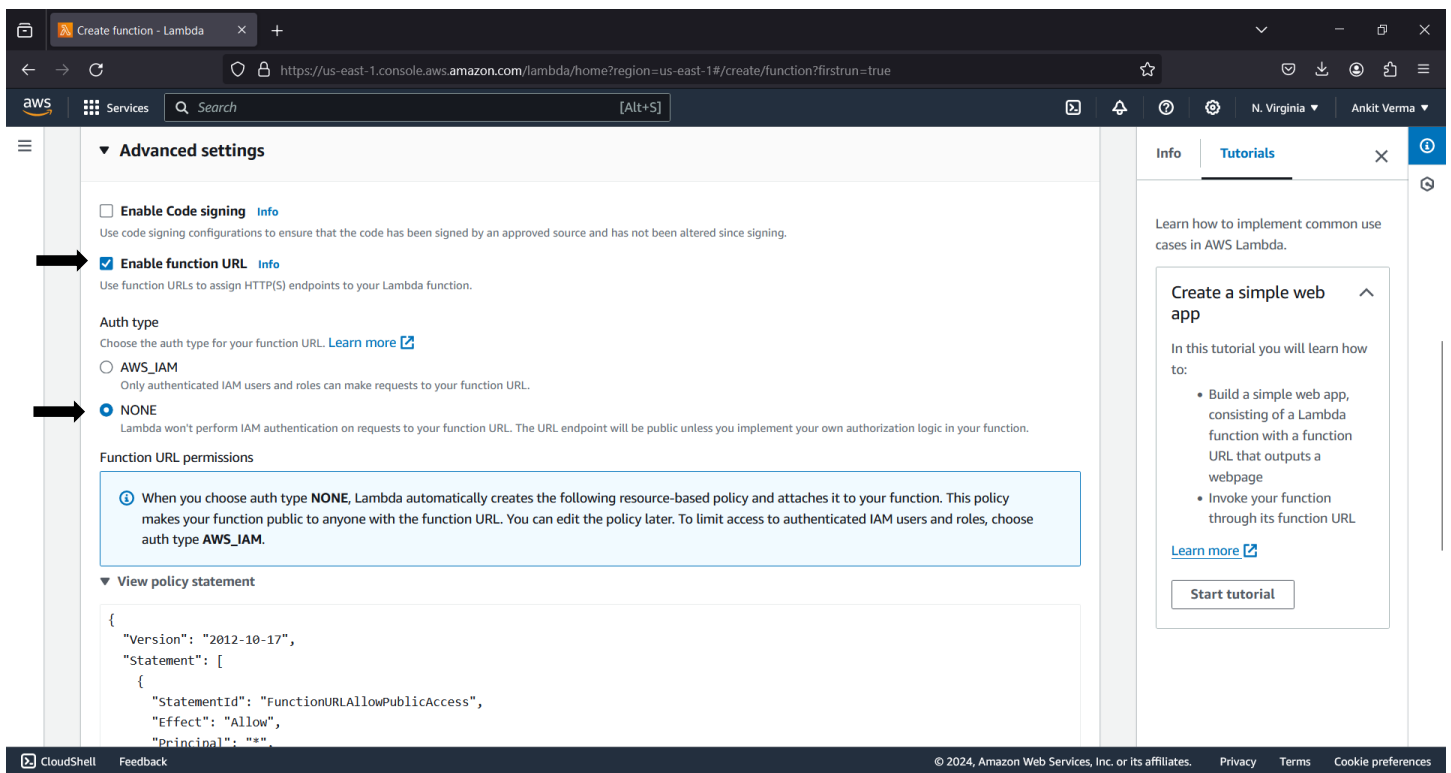
In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

### Step 4: Under the Advanced settings, check the Enable Function URL checkbox & under Auth type, select None then, Click on Create Function.



Create function - Lambda

Advanced settings

☐ Enable Code signing [Info](#)  
Use code signing configurations to ensure that the code has been signed by an approved source and has not been altered since signing.

☒ Enable function URL [Info](#)  
Use function URLs to assign HTTP(S) endpoints to your Lambda function.

**Auth type**  
Choose the auth type for your function URL. [Learn more](#)

☐ AWS\_IAM  
Only authenticated IAM users and roles can make requests to your function URL.

☒ NONE  
Lambda won't perform IAM authentication on requests to your function URL. The URL endpoint will be public unless you implement your own authorization logic in your function.

**Function URL permissions**

When you choose auth type **NONE**, Lambda automatically creates the following resource-based policy and attaches it to your function. This policy makes your function public to anyone with the function URL. You can edit the policy later. To limit access to authenticated IAM users and roles, choose auth type **AWS\_IAM**.

**View policy statement**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "StatementId": "FunctionURLAllowPublicAccess",
      "Effect": "Allow",
      "Principal": "*"
    }
  ]
}
```

Info

Tutorials

Learn how to implement common use cases in AWS Lambda.

### Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

Start tutorial

## Step 5: Click on the function URL to see the output.

The screenshot shows the AWS Lambda console for a function named 'Ankit'. The 'Function URL' field is highlighted with a black arrow. The URL is <https://chzy2qtnmurh5awagjcugupika0bdlitz.lambda-url.us-east-1.on.aws/>. The console also displays the function's ARN, last modified time, and code source.

Diagram Template

Ankit

Layers (0)

+ Add trigger

+ Add destination

Description

Last modified: 5 minutes ago

Function ARN: arn:aws:lambda:us-east-1:637423304400:function:Ankit

Function URL: <https://chzy2qtnmurh5awagjcugupika0bdlitz.lambda-url.us-east-1.on.aws/>

Code source

Test Deploy

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO: Implement
5     return {
6         'statusCode': 200,
7         'body': json.dumps('Hello from Ankit Verma!')}
8 }
```

## Step 6:

The screenshot shows a web browser with the function URL entered in the address bar: <https://chzy2qtnmurh5awagjcugupika0bdlitz.lambda-url.us-east-1.on.aws>.

"Hello from Ankit Verma!"