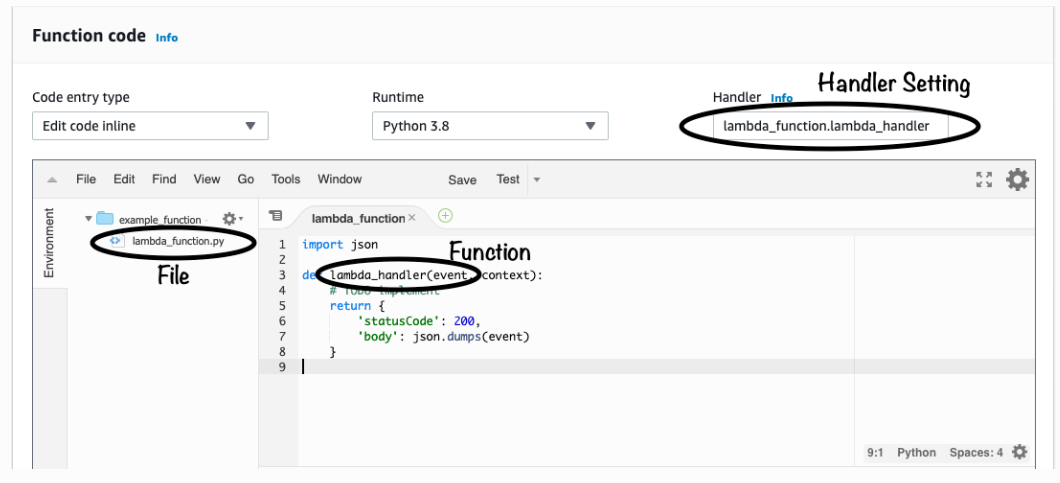
AWS Lambda is an Amazon serverless, full managed computing service that let's you to run the code. It is an event-driven computing service.

Let’s take a look of screenshot of lambda function from console.



**Handler/Handler function:**

The handler function is the starting point of your code. It's the python function that is executed when your lambda function runs.

By default handler will be --- lambda\_handler(event, context) but we can create our own handler.

**Creating Custom Handler:**

Open the lambda function and go to 'CODE' table then scroll down and go to EDIT.

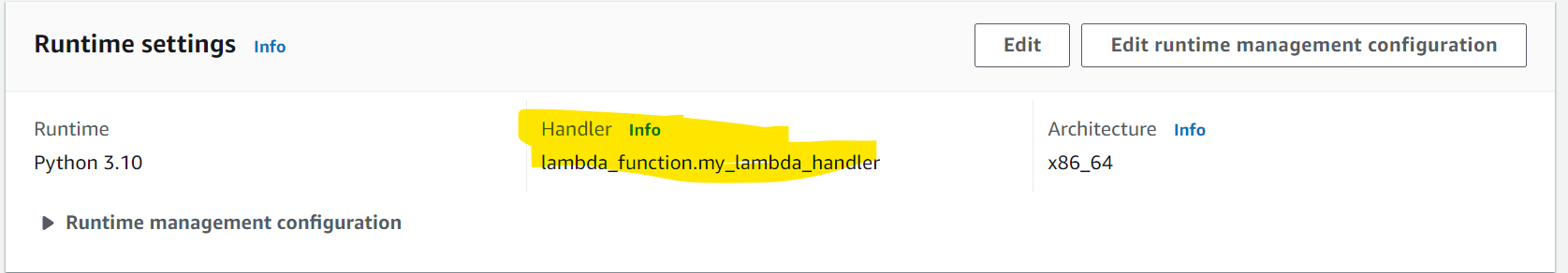
Here we can define our custom handler as --- handler\_file\_name.handler\_name

**Note:**

When we create custom handler/function, then make sure that handler/function is present in that file.

**Example**:

let say we have below handler configuration-



According to this configuration-

* Starting/entry point of aws lambda function will be ----my\_lambda\_handler(event, context ) function from lambda\_function.py file.
* Make sure that my\_lambda\_handler() is present in lambda\_function.py file.

**Context in AWS lambda**

Context object provides methods and properties that provide information about the invocation, function, and execution environment

**Creating lambda**

We can create AWS lambda many way as-

1. Using console
2. Using Cloud shell
3. Using CloudFormation **(CFT)**

<https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-lambda-function.html#cfn-lambda-function-ephemeralstorage>

**#############################################**

**# Creating Lambda using CloudFormation #--- Method 3rd**

**#############################################**

While creating lambda using CloudFormation template we can split the lambda creation in two part-

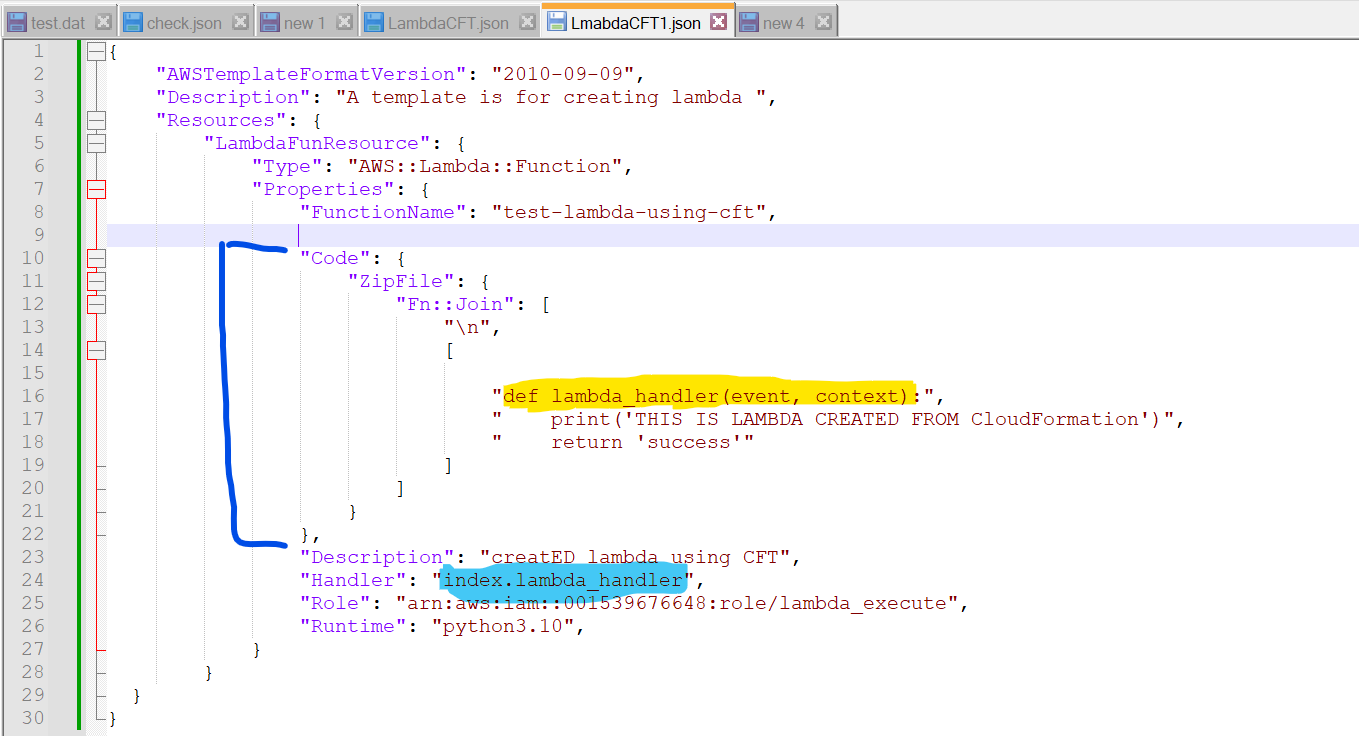
1. When code and dependency are packed in zip file and placed s3 bucket
2. Providing lambda function code in CloudFormation template

<https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-lambda-function.html#cfn-lambda-function-ephemeralstorage>

**Creating Lambda using CFT and code also in template -- Method 3rd (a)**

In below sample template code, it has lambda function code and lambda creation code both.





**Parameter description**

Runtime – programming language with version

Handler – name of file and method which will be entry point of our lambda (file\_name.method\_name).

Default file name – lambda\_function

Default function name – lambda\_handler(always give function name as lambda\_handler)

FunctionName – Name of lambda function by which it will be created

Code --- Code for the function

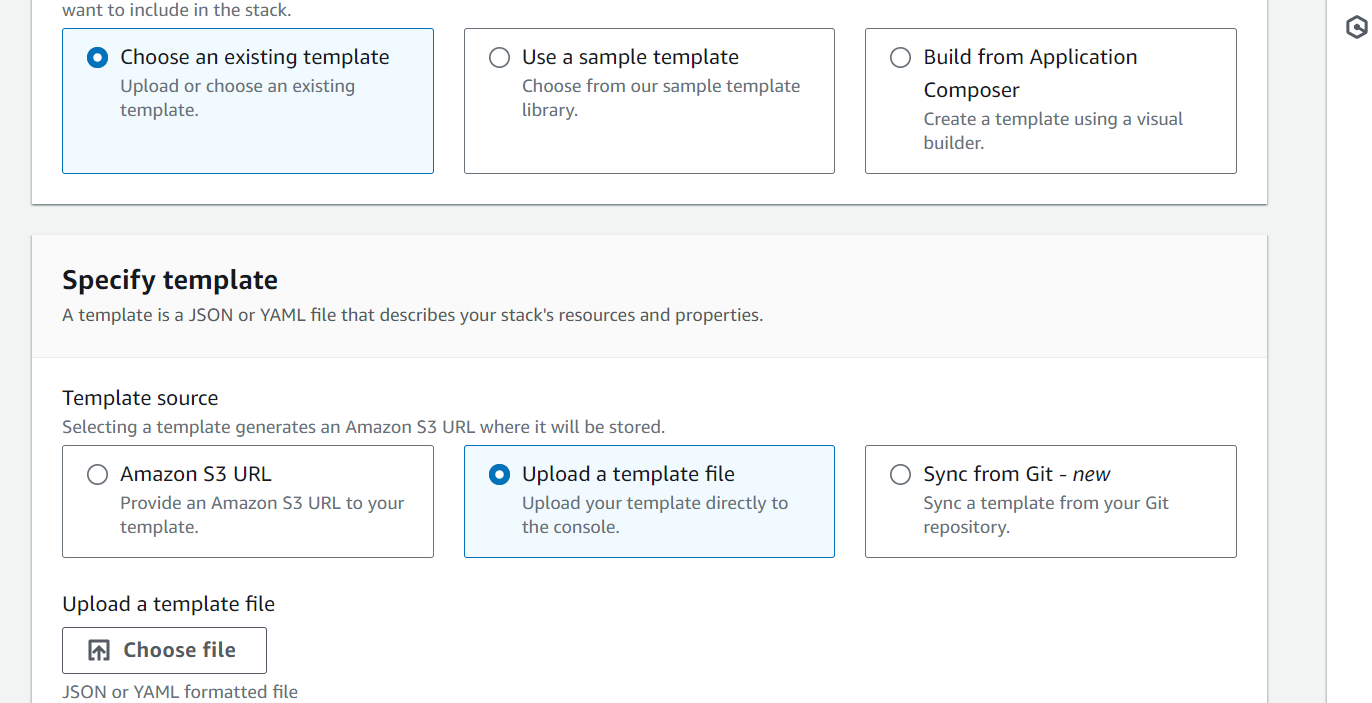
**Note:**

Based on above template file-

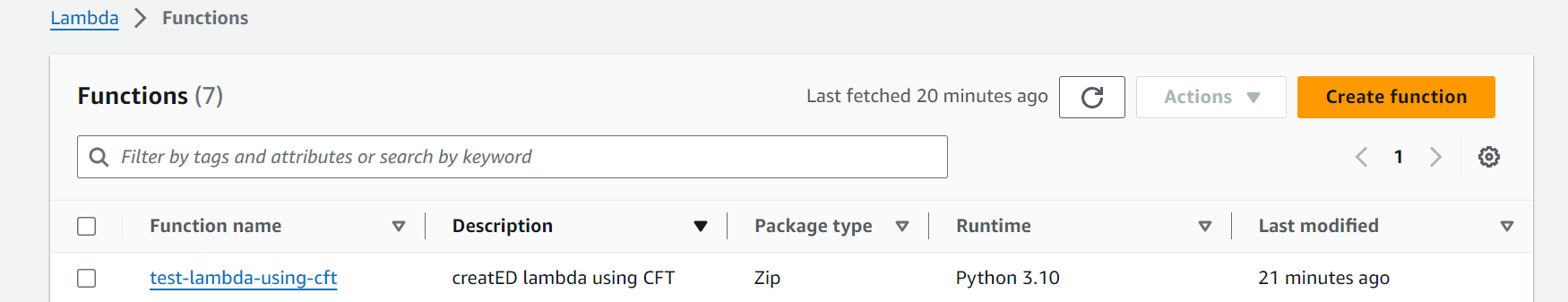
File name will be – index.py

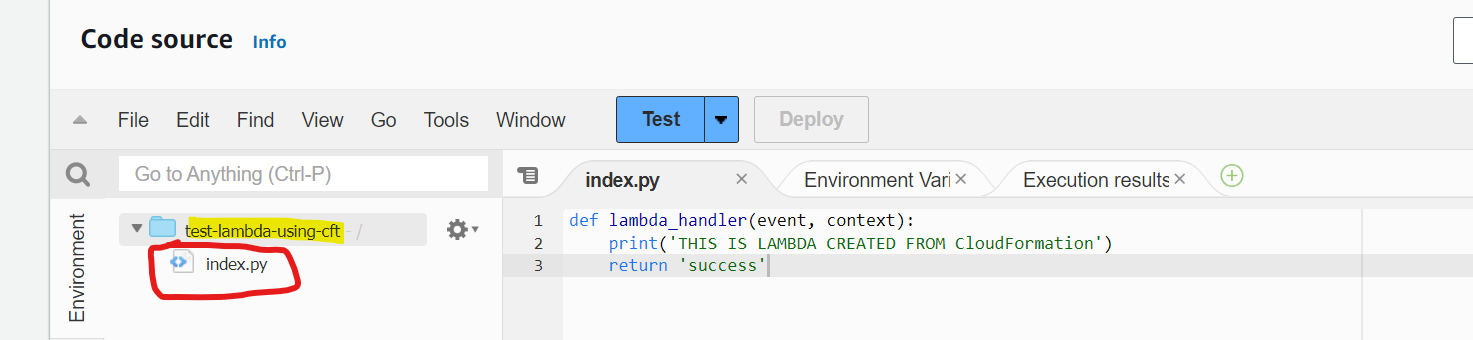
Function/metho name will be – lambda\_handler

Now upload this json file in cloud formation template and lambda function will get created.



Below screenshot shows the lambda created.





**Creating Lambda using CFT (code and dependency packed into zip ) --- Method 3 (b)**

**Step:**

* Create a folder locally on laptop / on cloud shell
* Create the lambda handler file you want, default name lambda\_function.py
* Create the lambda handler function in handler file

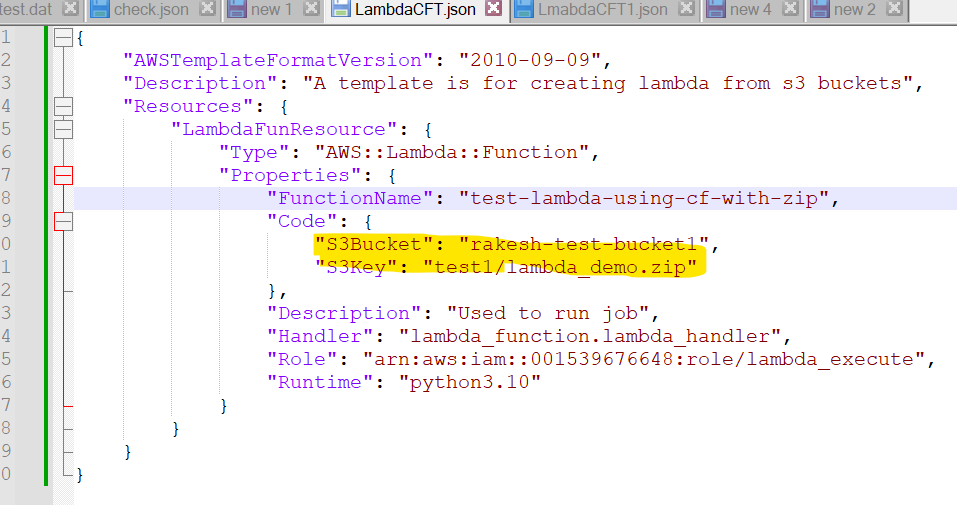
( optional – can be created after lambda is created using console)

* Install reqrered library using pip command (any of below method )

1. pip install --target \_path\_to\_install <library\_name>
2. Go to path where want to instaal then --- **pip install –target . <library\_name>**

* Zip the file and place that zip file in any bucket
* Now create the CFT json file and upload to CFT to create the lambda

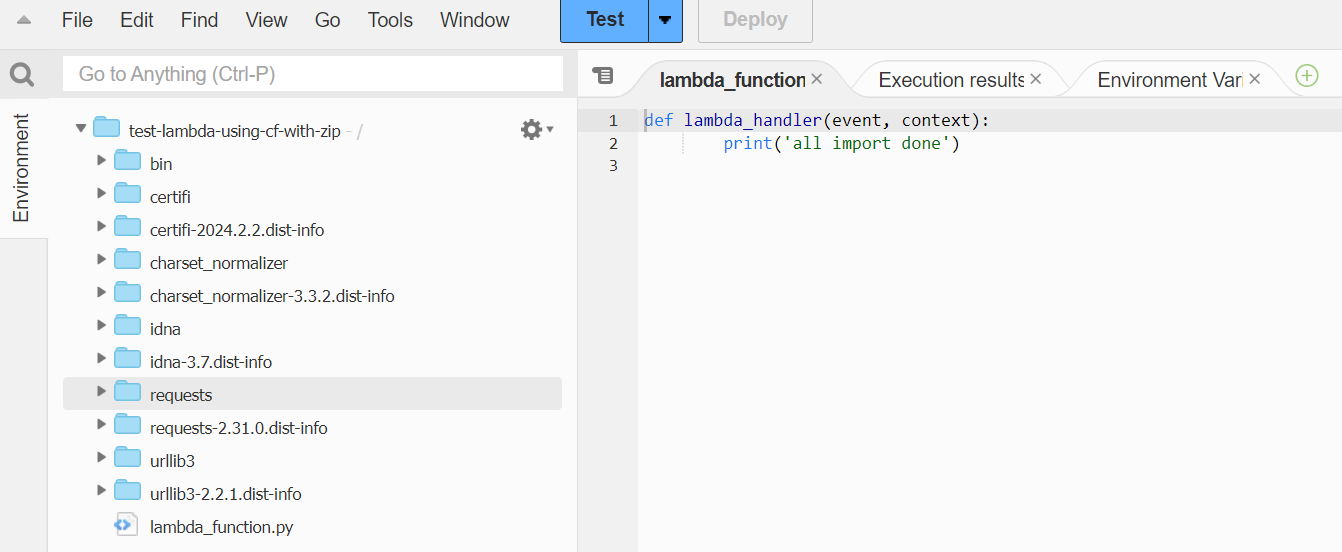




Here for learning purpose, I have zipped lambda\_demo folder which contains some libraries also which have lambda\_handler method in lambda\_function.py file as entry point as usual and placed in my bucket.

Add more detail how and what was in .zip folder and how it was created

Now upload this template and lambda function will be created as below-



**##############################**

**# Creating lambda using clous shell #**

**##############################**

This is exactly same as above one (Method 3rd(b)

**#######################################################################################**

**# AWS CodeBuild #**

**#######################################################################################**

**What is CodeBuild ?**

AWS CodeBuild is a fully managed build service that compiles source code, runs tests/unit tests, and produces software packages that are ready to deploy. With CodeBuild, you don’t need to worry about provisioning and managing your own build infrastructure. You simply provide your build project’s source code and build settings, and CodeBuild handles the rest.

**Some keywords used in CodeBuild**

Below are some common naming terminologies that we used in codebuild.

**Build project**

A build project includes information about how to run a build, including where to get the source code, which build environment to use, which build commands to run, and where to store the build output.

**Build environment**

A build environment represents a combination of operating system, programming language runtime, and tools that CodeBuild uses to run a build

**buildspec.yml**

A buildspec is a collection of build commands and related settings, in YAML format, that CodeBuild uses to run a build. You can include a buildspec as part of the source code or you can define a buildspec when you create a build project

For build syntax and phased check below weblink-

<https://docs.aws.amazon.com/codebuild/latest/userguide/build-spec-ref.html#build-spec-ref-name-storage>

**Buildspec file name and storage location**

1. If you include a buildspec as part of the source code, by default, the buildspec file must be named buildspec.yml and placed in the root of your source directory.
2. You can specify only one buildspec for a build project, regardless of the buildspec file's name.
3. You can override the default buildspec file name and location. For example, you can:

* Use a different buildspec file for different builds in the same repository, such as buildspec\_debug.yml and buildspec\_release.yml.

Store a buildspec file somewhere other than the root of your source directory, such as config/buildspec.yml or in an S3 bucket.

The S3 bucket must be in the same AWS Region as your build project. Specify the buildspec file using its ARN (for example, arn:aws:s3:::<my-codebuild-sample2>/buildspec.yml).

Check more on below link-

<https://docs.aws.amazon.com/codebuild/latest/userguide/build-spec-ref.html>

**#######################################################################################**

**# AWS CodeBuild #**

**#######################################################################################**

**What is CodeDeploy**

* AWS CodeDeploy is a deployment service that automates application deployments to Amazon EC2 instances, on-premises instances, serverless Lambda functions, or Amazon ECS services.
* CodeDeploy can deploy application content that runs on a server and is stored in Amazon S3 buckets, GitHub repositories, or Bitbucket repositories.
* CodeDeploy can also deploy a serverless Lambda function. You do not need to make changes to your existing code before you can use CodeDeploy.

<https://docs.aws.amazon.com/codedeploy/latest/userguide/welcome.html>