

Step1. Create One AWS lambda function with below configuration

The screenshot shows the 'Create function' page in the AWS Lambda console. At the top, there are four options to create a function: 'Author from scratch' (selected), 'Use a blueprint', 'Container image', and 'Browse serverless app repository'. Below these is the 'Basic information' section. The 'Function name' field contains 'CDN-Invalidation'. The 'Runtime' dropdown is set to 'Node.js 14.x'. The 'Architecture' dropdown is set to 'x86_64'. The 'Permissions' section shows the default role. At the bottom, there is a link to 'Change default execution role'.

Create function [Info](#)

Choose one of the following options to create your function.

Author from scratch [Info](#)
Start with a simple Hello World example.

Use a blueprint [Info](#)
Build a Lambda application from sample code and configuration presets for common use cases.

Container image [Info](#)
Select a container image to deploy for your function.

Browse serverless app repository [Info](#)
Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name [Info](#)
Enter a name that describes the purpose of your function.
CDN-Invalidation
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.
Node.js 14.x

Architecture [Info](#)
Choose the instruction set architecture you want for your function code.
☒ x86_64
☐ arm64

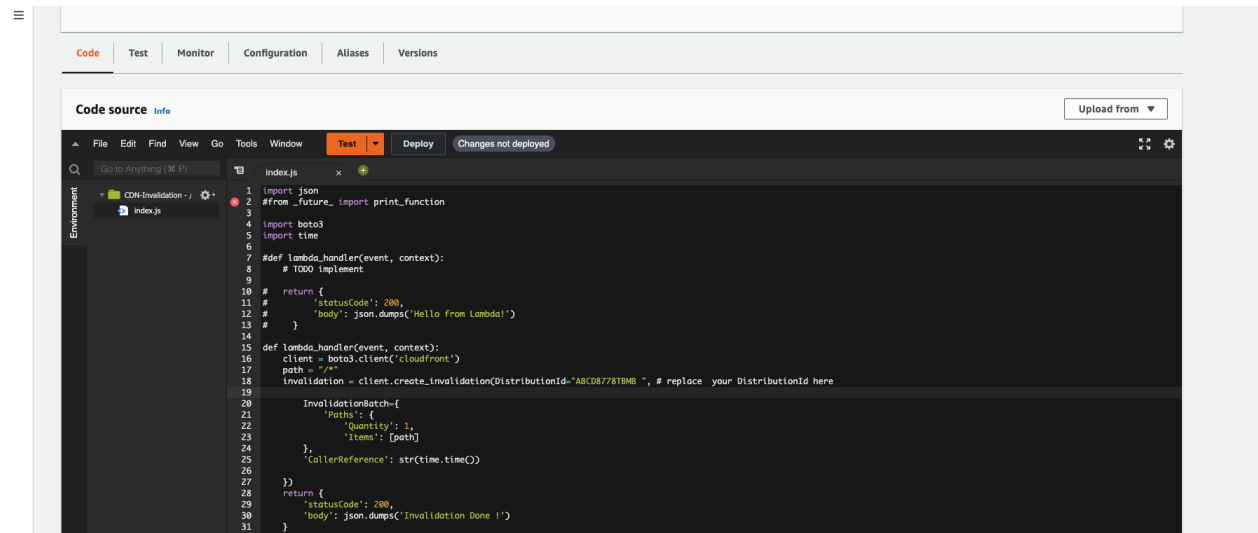
Permissions [Info](#)
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

[Change default execution role](#)

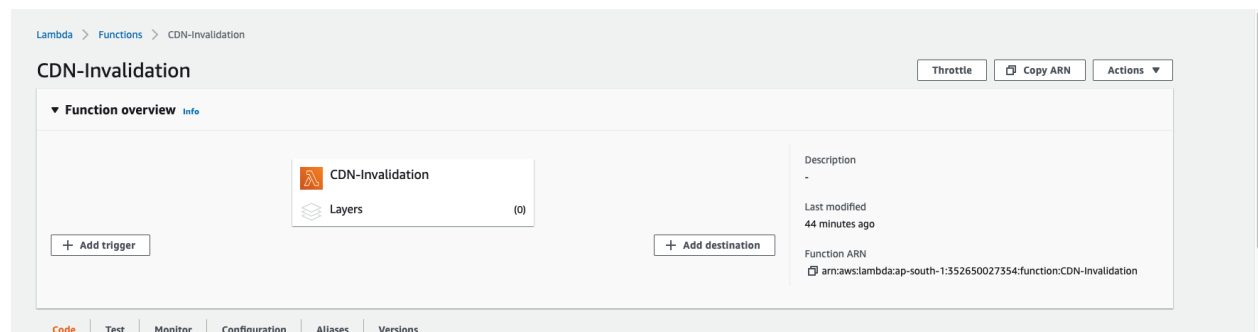
Step2. Add below code in index.js

```
import json
import boto3
import time
def lambda_handler(event, context):
    client = boto3.client('cloudfront')
    path = "/"
    invalidation = client.create_invalidation(DistributionId="ABCD8778TBMB ", # replace your
DistributionId here
        InvalidationBatch={
            'Paths': {
                'Quantity': 1,
                'Items': [path]
            },
            'CallerReference': str(time.time())
        })
    return {
        'statusCode': 200,
        'body': json.dumps('Invalidation Done !')
    }
```

Note: select python3.8 environment for above function.



Step3. Add Trigger to this function from Add Trigger button



Step4. Create API Gateway for

Trigger configuration



API Gateway

api application-services aws serverless



Add an API to your Lambda function to create an HTTP endpoint that invokes your function. API Gateway supports two types of RESTful APIs: HTTP APIs and REST APIs. [Learn more](#)

API

Create a new API or attach an existing one.

Create an API



API type



HTTP API

Create an HTTP API.



REST API

Create a REST API.

Security

Configure the security mechanism for your API endpoint.

Open



Additional settings

API name

Choose a name for your API. API names don't need to be unique.

CDN-Invalidation-API

Deployment stage

The name of your API's deployment stage.

default

Step 5: Lambda IAM Role modification

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudfront:CreateInvalidation"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Create cloudfornt invalidation policy and attach to the IAM role attached to the lambda function.