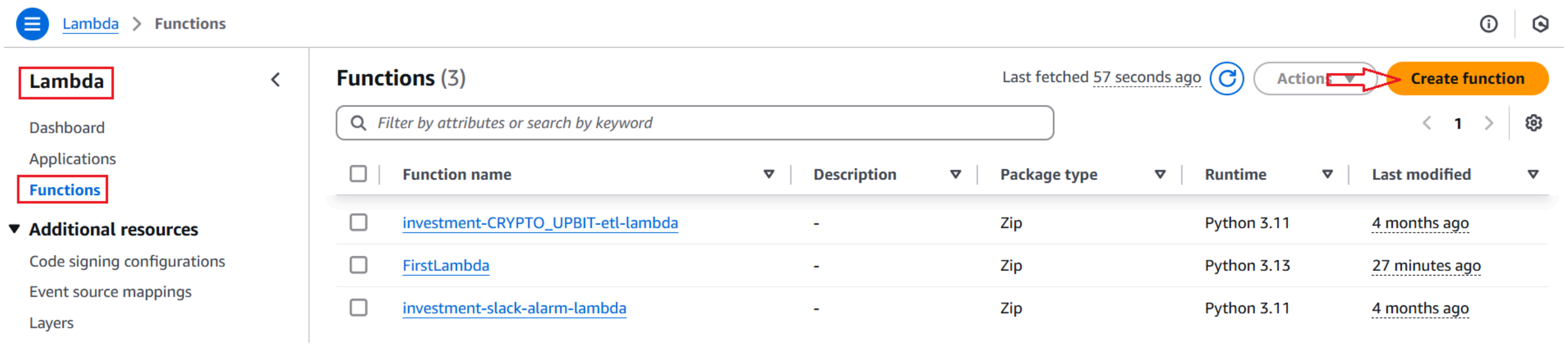
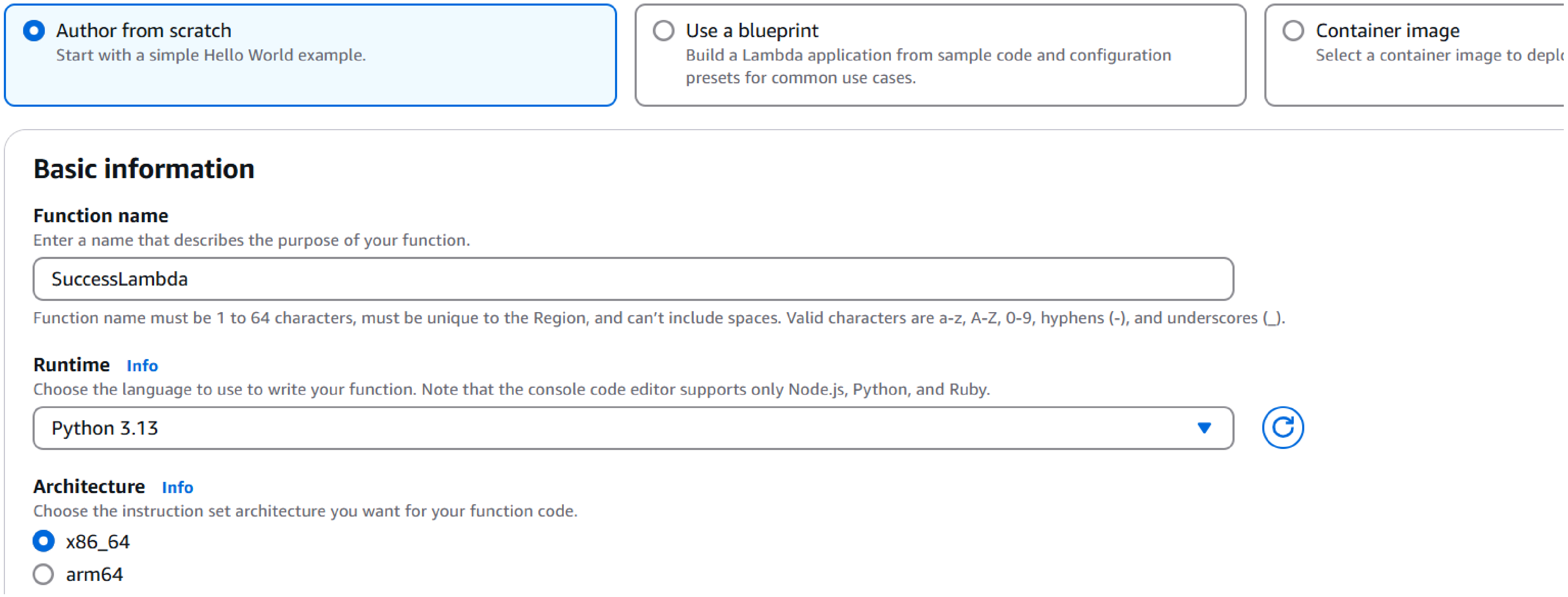
SuccessLambda 생성

단계1: Create function



Function name	Description	Package type	Runtime	Last modified
investment-CRYPTO_UPBIT-etl-lambda	-	Zip	Python 3.11	4 months ago
FirstLambda	-	Zip	Python 3.13	27 minutes ago
investment-slack-alarm-lambda	-	Zip	Python 3.11	4 months ago



[✓ 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 depl

Basic information

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

SuccessLambda

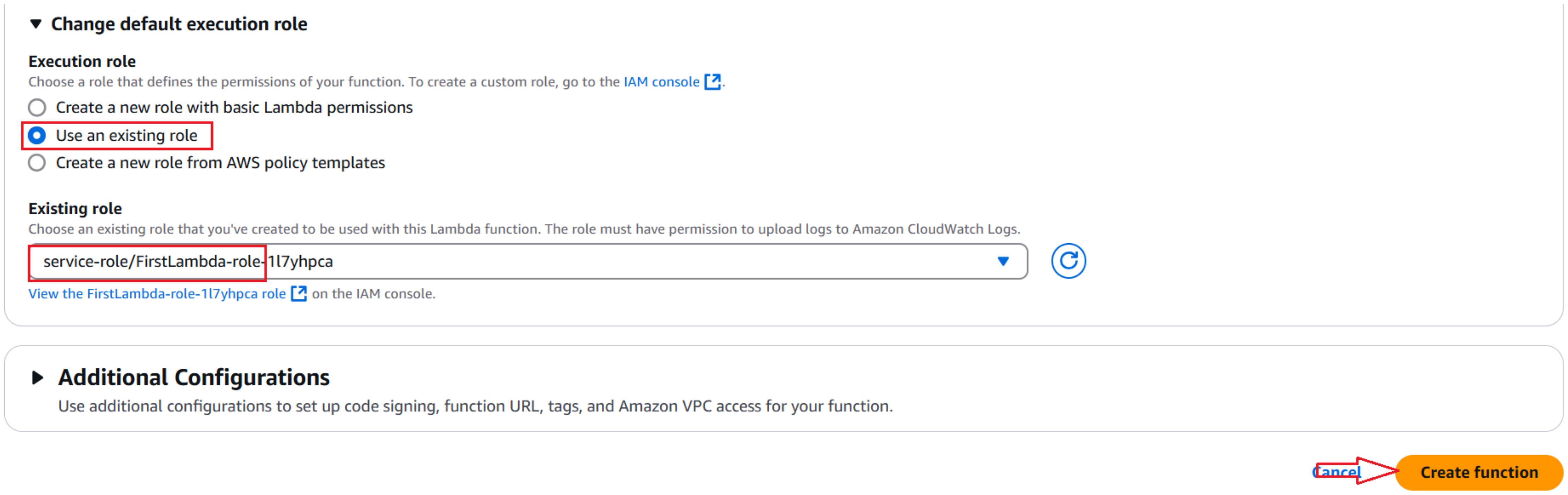
Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

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

Python 3.13

Architecture Info
Choose the instruction set architecture you want for your function code.

[✓ x86_64]
arm64



▼ Change default execution role

Execution role

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

- ○ Create a new role with basic Lambda permissions
- [✓ 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.

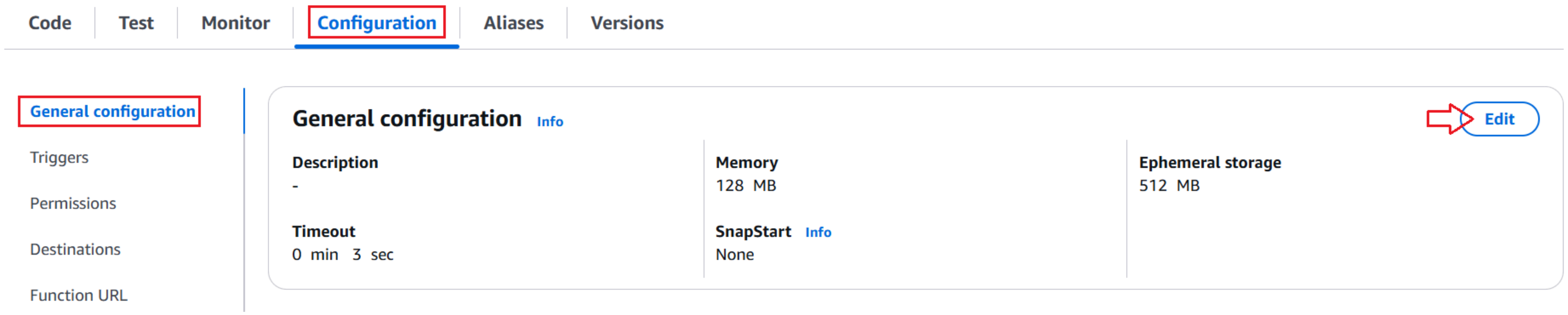
service-role/FirstLambda-role-1l7yhpca

View the FirstLambda-role-1l7yhpca role on the IAM console.

▶ **Additional Configurations**

Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.

단계2: configuration > Timeout 수정



General configuration

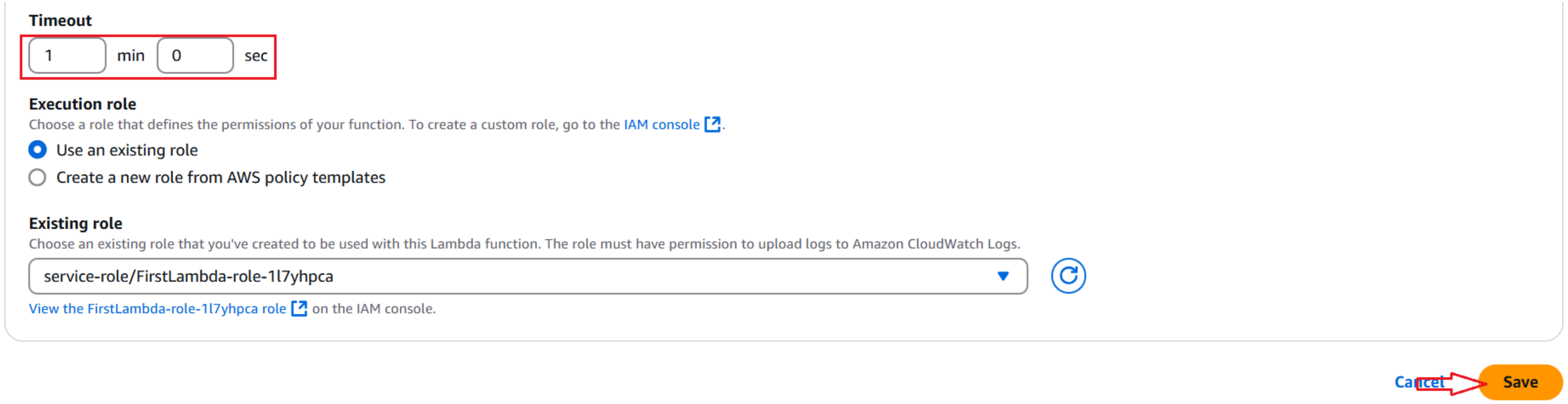
Description
-

Memory
128 MB

Ephemeral storage
512 MB

Timeout
0 min 3 sec

SnapStart
None


Timeout

1 min 0 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]

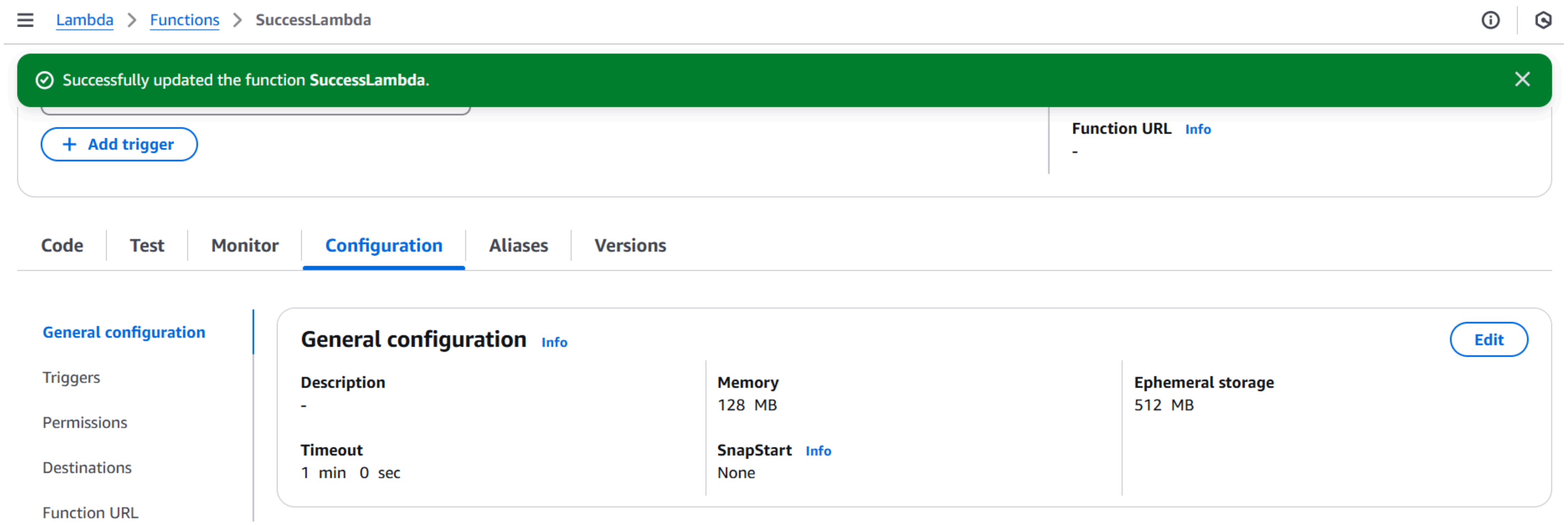
Create a new role from AWS policy templates

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/FirstLambda-role-1l7yhpca

View the FirstLambda-role-1l7yhpca role on the IAM console.


Lambda > Functions > SuccessLambda

Successfully updated the function **SuccessLambda**.

Add trigger

Function URL Info
-

General configuration Info

Description
-

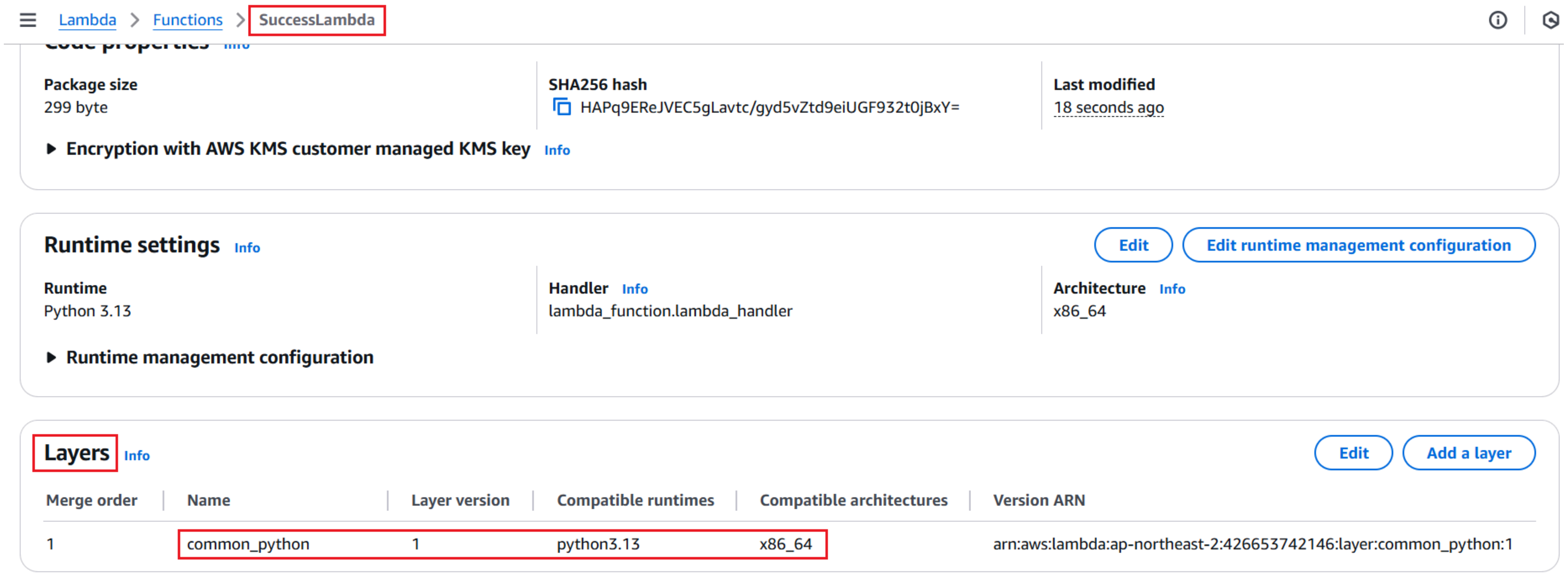
Memory
128 MB

Ephemeral storage
512 MB

Timeout
1 min 0 sec

SnapStart Info
None

단계3: Layer 추가



Lambda > Functions > SuccessLambda

Package size	SHA256 hash	Last modified
299 byte	HAPq9EReJVEC5gLavtc/gyd5vZtd9eiUGF932t0jBxY=	18 seconds ago

▶ Encryption with AWS KMS customer managed KMS key Info

Runtime settings Info

Edit | Edit runtime management configuration

Runtime	Handler	Architecture
Python 3.13	lambda_function.lambda_handler	x86_64

▶ Runtime management configuration

Layers Info

Edit | Add a layer

Merge order	Name	Layer version	Compatible runtimes	Compatible architectures	Version ARN
1	common_python	1	python3.13	x86_64	arn:aws:lambda:ap-northeast-2:426653742146:layer:common_python:1

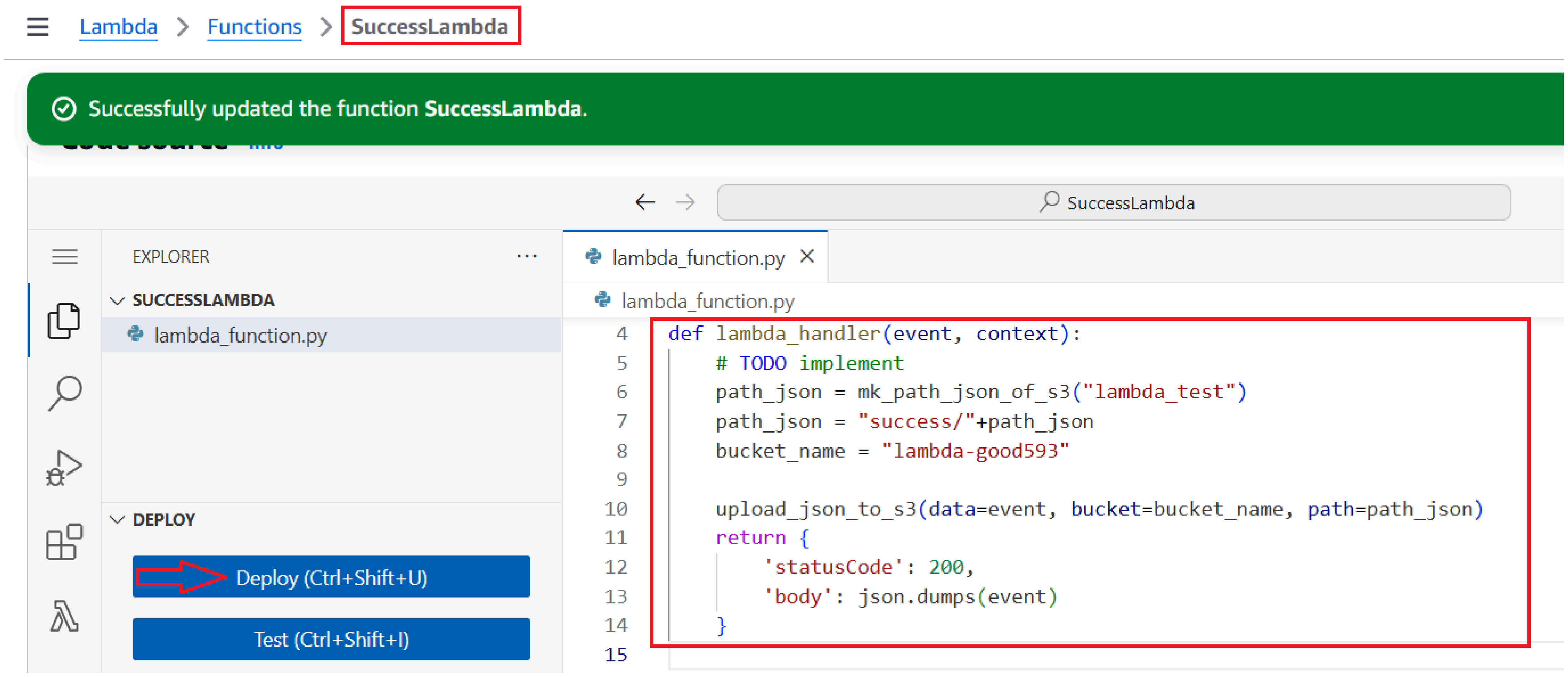
단계4: 코드 수정

```python
import json
from common.aws_s3 import mk_path_json_of_s3, upload_json_to_s3, download_json_from_s3

def lambda_handler(event, context):
    # TODO implement
    path_json = mk_path_json_of_s3("lambda_test")
    path_json = "success/"+path_json
    bucket_name = "lambda-good593"

    upload_json_to_s3(data=event, bucket=bucket_name, path=path_json)
    return {
        'statusCode': 200,
        'body': json.dumps(event)
    }
```

단계5: Deploy



```python
def lambda_handler(event, context):
    # TODO implement
    path_json = mk_path_json_of_s3("lambda_test")
    path_json = "success/"+path_json
    bucket_name = "lambda-good593"

    upload_json_to_s3(data=event, bucket=bucket_name, path=path_json)
    return {
        'statusCode': 200,
        'body': json.dumps(event)
    }
```