

...



laC;

# Infrastructure as a Code

+CloudFormation과 SAM 실습

# Agenda

## 발표자 소개

## 주재빈

E-MAIL: [jbinchoo@gmail.com](mailto:jbinchoo@gmail.com)

2019년 차세대분산시스템 수강  
2021학년도 건국대학교 컴퓨터공학과 졸업

졸업 후 소프트웨어 QA에 임하다가  
클라우드 및 솔루션 아키텍트 쪽 직무 기회를 찾아가는 중입니다.

현재 한 카카오톡 챗봇 서비스를 개발&운영하고 있습니다.



[binchoo](#)



[카카오톡 챗봇, 여행 비서 페이몬!](#)



## 1. 코드형 인프라

- IaC 개요
- CloudFormation 소개
- CloudFormation 자원 생성 흐름
- 데모1: S3 버킷 만들기

## 2. CloudFormation Template

- 템플릿 파일의 구조
- 데모2: SNS 연동 람다 만들기

## 3. CloudFormation & SAM 실습

- AWS SAM 소개
- 데모3: SAM 활용 SNS 연동 람다 만들기
- DIY: 유입경로 분석 파이프라인 만들기



# 준비 사항



- [AWS SAM CLI](#) 설치하기  
⚠ [AWS CLI](#)에 의존성이 있습니다.
- AWS CLI에 자신의 IAM User 크레덴셜 설정하기  
→ [aws configure 명령어](#)
- Python 3.7, 3.8, 3.9 중 하나
- 실습 소스코드 다운로드
- YAML 작성에 도움이 되는 에디터 (필수 아님)



# 실습 소스코드 및 교안

- <https://github.com/binchoo/2022-dms-iac>
- src 디렉토리 → 실습 소스코드
- ppt 디렉토리 → 강의 교안

# 1. 코드형 인프라

# Power of the console

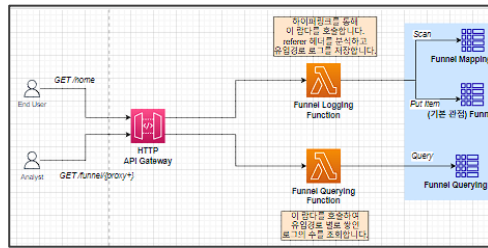
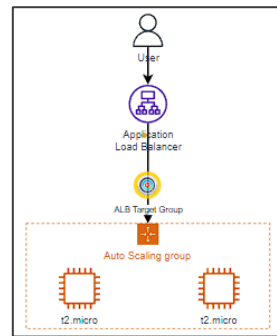


콘솔 홈 정보 기본 레이아웃으로 재설정 + 위젯 추가

새 위젯 애플리케이션(들) 소개합니다. 콘솔 홈의 하단에서 찾을 수 있습니다.

최근에 방문한 서비스 정보

CloudWatch	API Gateway
Lambda	IAM
Trusted Advisor	Elastic Beanstalk
EC2	S3
AWS Budgets	CloudFormation
AWS Cost Explorer	Route 53
DynamoDB	Certificate Manager



# Shall I repeat??

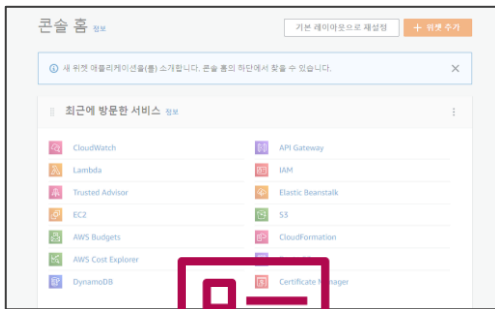


“100개의 계정에  
똑같은 환경을 구축해야 해.”

(eg. 다중 리전 구축)

수작업으로 반복 구축하기 어렵습니다!

# Infrastructure as a Code(1)



인프라를  
명세하는 코드



EC2



Lambda

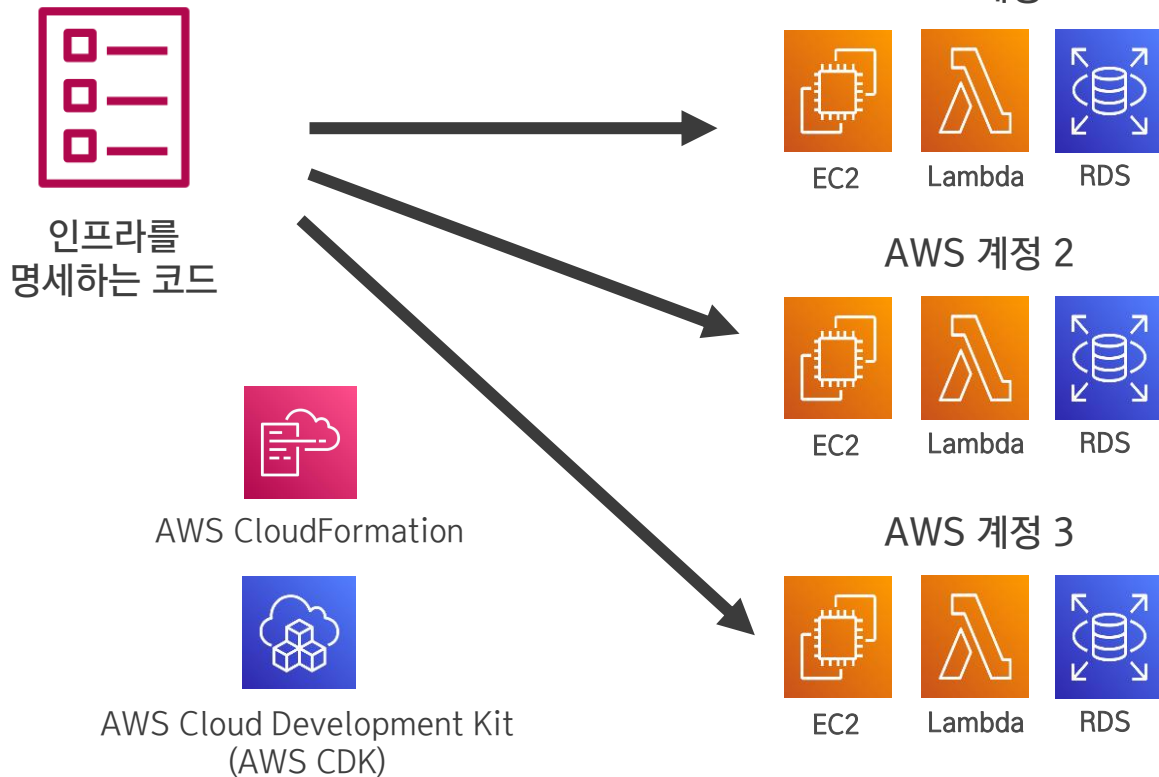


RDS

- 코드를 통한 인프라 할당과 관리
- 사람이 읽기 좋고, 머신이 해석할 수 있는 파일



# Infrastructure as a Code(2)



# IaC Benefits



# AWS CloudFormation



- IaC service for AWS
- Free of charge
  - Pay for the provisioned infrastructures
- **Template** to spin up AWS infrastructure
  - Written in JSON or YAML
  - Quickly replicate your infrastructure
    - Multiple accounts
    - Dev, stage, prod environments
    - Disaster recovery
    - etc.

# AWS CloudFormation

## Concepts

- Template
  - Blueprints for AWS resources
  - JSON or YAML format
- Stack
  - created by submitting templates
  - can be created, updated and deleted
- Change set
  - summary of proposed changes in stack by submitting a template
  - how the change will impact resources



Template



Stack



Change set

# Sample Template(1) – S3 Bucket

```
1  AWSTemplateFormatVersion: 2010-09-09
2  Resources:
3    S3Bucket:
4      Type: AWS::S3::Bucket
5  Outputs:
6    BucketName:
7      Value: !Ref S3Bucket
8      Description: Name of the sample Amazon S3 bucket.
9
```

A single bucket is not a big deal.

...

# Sample Template(2) – SNS & Lambda

```
AWSTemplateFormatVersion: '2010-09-09'
Description: >
  sns-lambda-template

Example CloudFormation template to subscribe a lambda to an SNS topic.
Resources:
  PatientTopic:
    Type: AWS::SNS::Topic
    Properties:
      DisplayName: patient-topic
      TopicName: patient-topic
      Subscription:
        - Protocol: lambda
          Endpoint: !GetAtt PatientCheckoutFunction.Arn

  PatientTopicPolicy:
    Type: AWS::SNS::TopicPolicy
    Properties:
      Topics:
        - !Ref PatientTopic
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Action: 'sns:Publish'
            Resource: !Ref PatientTopic
            Principal:
              AWS: '*'
            Condition:
              ArnLike:
                AWS:SourceArn: !Sub 'arn:aws:*:*:${AWS::AccountId}:*'

  PatientCheckoutFunction:
```

```
    Type: AWS::Lambda::Function
    Properties:
      Runtime: python3.8
      Handler: index.handler
      Code:
        ZipFile: |
          import json
          def handler(event, context):
            patients = [{ 'name': json.loads(record['Sns']['Message'])['name'], 'st
            print(patients)
      Role: !GetAtt PatientCheckoutFunctionExecutionRole.Arn

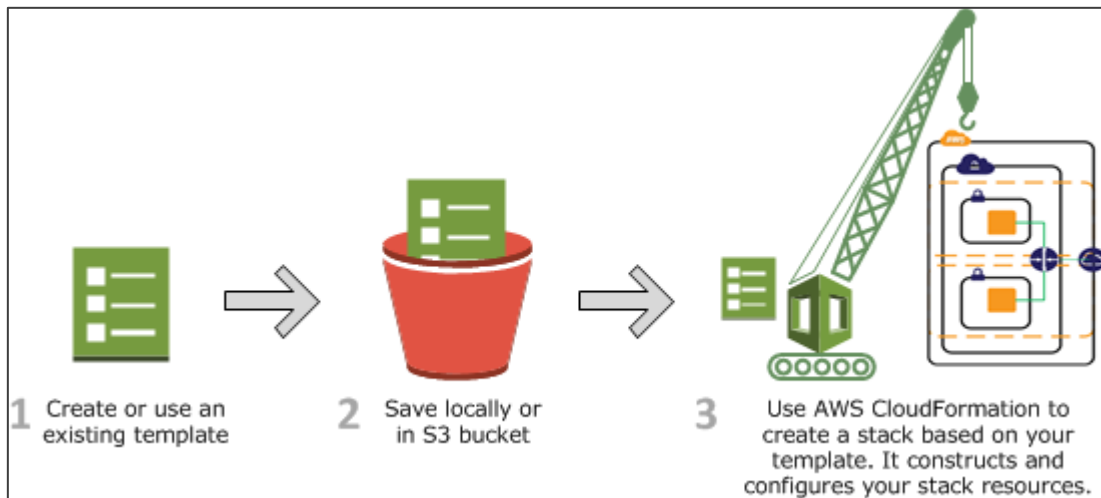
  PatientCheckoutFunctionInvokePermission:
    Type: AWS::Lambda::Permission
    Properties:
      Action: 'lambda:InvokeFunction'
      FunctionName: !Ref PatientCheckoutFunction
      Principal: sns.amazonaws.com

  PatientCheckoutFunctionExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - lambda.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: "/"
      Policies:
        - PolicyName: root
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Effect: Allow
```

A lambda subscribing to an SNS topic

# CloudFormation Flow

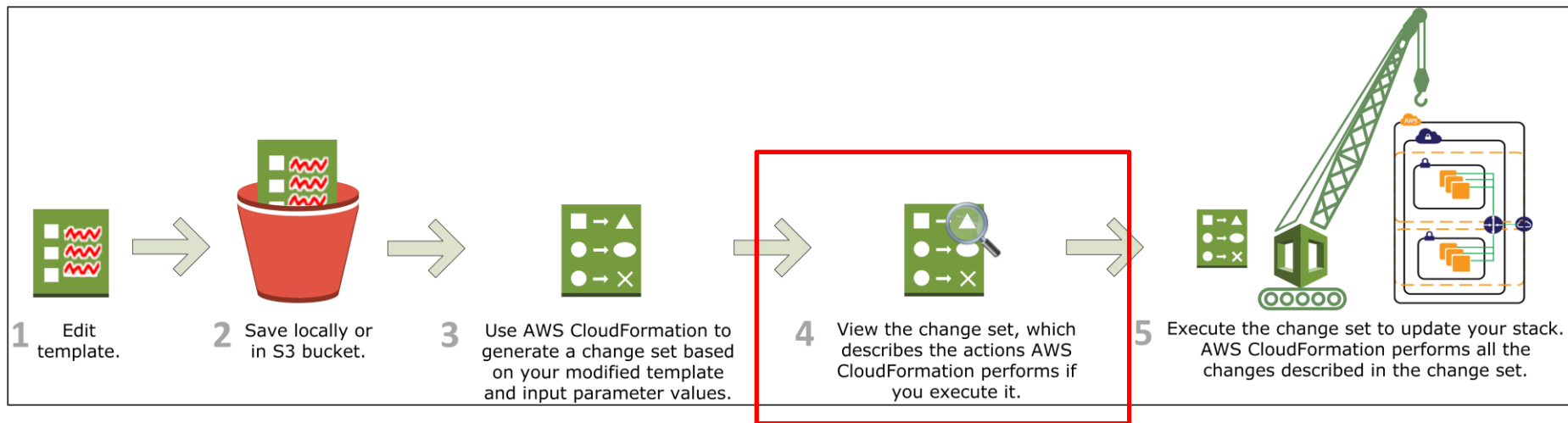
## CFN의 자원 생성 흐름



- 1) 템플릿에 자원 명세를 작성하고
- 2) S3 버킷에 업로드하고
- 3) CloudFormation에 제출하면 CFN 스택이 만들어진다.

# CloudFormation Flow

CFN의 자원 생성 흐름 – Change Sets 검토할 시



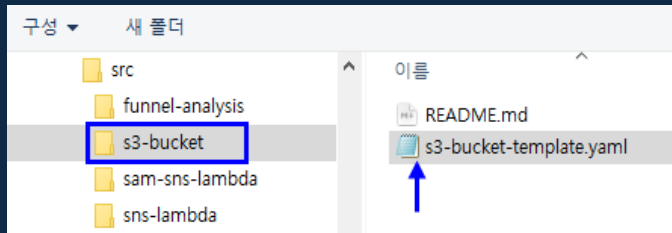
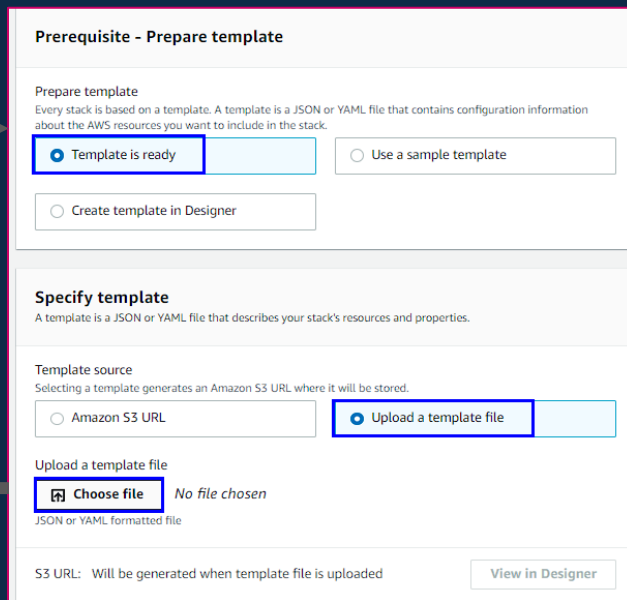
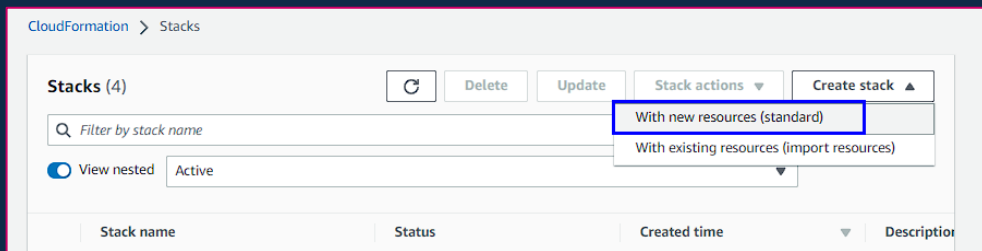
※ 관리자가 손수 Change Set을 검토하는 과정이 추가된다.



# CloudFormation Demo①

## Submitting a CFT that creates an s3 bucket via CFN console

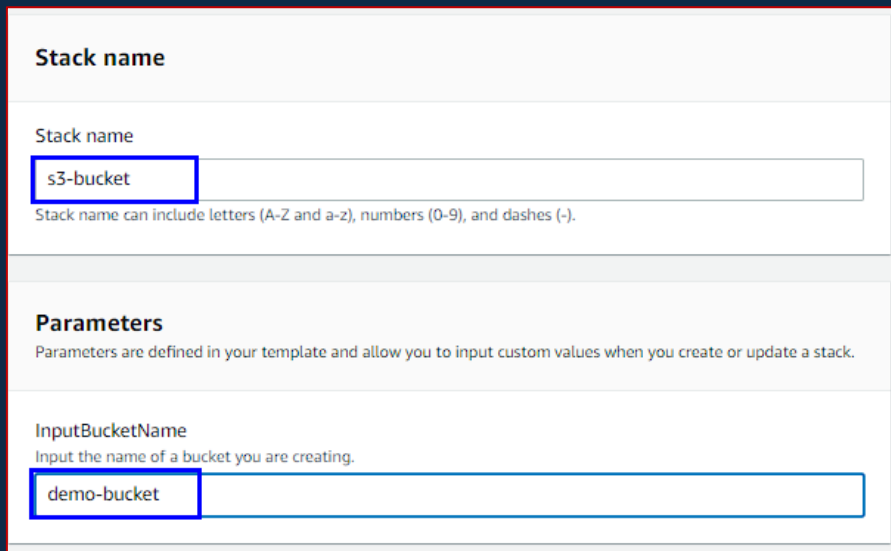
1. Download the template @ <https://github.com/binchoo/2022-dms-iac/tree/master/src/s3-bucket>
2. Submit the `s3-bucket-template.yaml` file via the AWS CloudFormation console.



# CloudFormation Demo①

## Submitting a CFT that creates an s3 bucket via CFN console

3. Set the name of the CloudFormation stack. I set it to `s3-bucket`.
4. You can change the name of the s3 bucket, if necessary, giving other values to the `InputBucketName` parameter.



The screenshot shows the AWS CloudFormation console interface. It is divided into two main sections: 'Stack name' and 'Parameters'. In the 'Stack name' section, there is a text input field containing 's3-bucket'. Below the field, a small note states: 'Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-)'. In the 'Parameters' section, there is a text input field for the 'InputBucketName' parameter, which contains the value 'demo-bucket'. The entire form is outlined with a red border.

**Stack name**

Stack name

s3-bucket

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

**Parameters**

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

InputBucketName

Input the name of a bucket you are creating.

demo-bucket

# CloudFormation Demo①

## Submitting a CFT that creates an s3 bucket via CFN console

5. Finally, submit your CFT.
6. Verify that the stack outputs tap is displaying the bucket name of your new s3 bucket.

The screenshot shows the AWS CloudFormation console. The breadcrumb navigation at the top reads 'CloudFormation > Stacks > s3-bucket'. On the left sidebar, under 'Stacks (5)', the 's3-bucket' stack is listed with a status of 'CREATE\_COMPLETE'. The main panel shows the 's3-bucket' stack with tabs for 'Stack info', 'Events', 'Resources', 'Outputs', 'Parameters', and 'Templ'. The 'Outputs' tab is selected and highlighted with a blue box. Below the tabs, there is a search bar for outputs. A table displays the stack's outputs:

Key	Value	Description	Export name
BucketName	ap-northeast-2-305992497901-demo-bucket	Name of the sample Amazon S3 bucket.	-



## 2. CloudFormation 템플릿



# JSON vs. YAML



## JSON

ECMAScript 3판이 정의한 JS 객체 리터럴 표현법

```
1 {  
2   "AWSTemplateFormatVersion": "2010-09-09",  
3   "Parameters" : {  
4     "SNSEmail" : {  
5       "Type" : "String",  
6       "Description" : "Enter email for SNS notification"  
7     }  
8   },  
9   "Resources" : {  
10    "AlertSNSTopic" : {  
11      "Type" : "AWS::SNS::Topic",  
12      "Properties" : {  
13        "Subscription" : [{  
14          "Endpoint" : { "Ref" : "SNSEmail" },  
15          "Protocol" : "email"  
16        }]  
17      }  
18    }  
19  }  
20 }
```

## YAML

인간 친화적, 언어 무관, 유니코드 기반 직렬화 표현법

```
1   AWSTemplateFormatVersion: '2010-09-09'  
2   Parameters:  
3     SNSEmail:  
4       Type: String  
5       Description: Enter email for SNS notification  
6   Resources:  
7     AlertSNSTopic:  
8       Type: 'AWS::SNS::Topic'  
9       Properties:  
10        Subscription:  
11          -  
12            Endpoint:  
13              Ref: SNSEmail  
14            Protocol: email
```

NOT A BIG DEAL



# CFT Anatomy

AWSTemplateFormatVersion: *"version date"*

Description:

*String*

Metadata:

*template metadata*

Parameters:

*set of parameters*

Rules:

*set of rules*

Mappings:

*set of mappings*

Conditions:

*set of conditions*

Transform:

*set of transforms*

Resources:

*set of resources*

Outputs:

*set of outputs*

- ①AWSTemplateFormatVersion
- ②Description
- Metadata
- ③Parameters
- Rules
- Mappings
- Conditions
- ④Resources (*required*)
- ⑤Outputs
- Transform

이 교안은 빨간색 영역들을 소개하지만  
풍부한 표현의 CFT를 작성하려면 나머지 내용들도 알아야 합니다. 🙏

# CFT Anatomy

## ①Template Version

```
{  
  "AWSTemplateFormatVersion" : "2010-09-09",  
  "Description" : "A simple EC2 instance",  
  "Resources" : {  
    "MyEC2Instance" : {  
      "Type" : "AWS::EC2::Instance",  
      "Properties" : {  
        "ImageId" : "ami-0ff8a91507f77f867",  
        "InstanceType" : "t1.micro"  
      }  
    }  
  }  
}
```

임의의 날짜를 적지 않습니다.  
"2010-09-09"를 유효하게 사용 가능합니다.

# CFT Anatomy

## ②Description

```
{  
  "AWSTemplateFormatVersion" : "2010-09-09",  
  "Description" : "A simple EC2 instance",  
  "Resources" : {  
    "MyEC2Instance" : {  
      "Type" : "AWS::EC2::Instance",  
      "Properties" : {  
        "ImageId" : "ami-0ff8a91507f77f867",  
        "InstanceType" : "t1.micro"  
      }  
    }  
  }  
}
```

템플릿이 명세하는 인프라에 대해 자유롭게 서술하면 됩니다.



# CFT Anatomy

## ③Parameters

- Enables us to input custom values

Parameters:

*ParameterLogicalID:*

*Type: DataType*

*ParameterProperty: value*

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Description" : "A simple EC2 instance",
  "Resources" : {
    "MyEC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : "ami-0ff8a91507f77f867",
        "InstanceType" : "t1.micro"
      }
    }
  }
}
```

☹ 값을 외부에서 주입하면 더 유연해질텐데?

# CFT Anatomy

## Input an EC2 instance type

```
"Parameters" : {  
  "InstanceTypeParameter" : {  
    "Type" : "String",  
    "Default" : "t2.micro",  
    "AllowedValues" : ["t2.micro", "m1.small", "m1.large"],  
    "Description" : "Enter t2.micro, m1.small, or m1.large. Default is t2.micro."  
  }  
}
```

```
"Ec2Instance" : {  
  "Type" : "AWS::EC2::Instance",  
  "Properties" : {  
    "InstanceType" : { "Ref" : "InstanceTypeParameter" },  
    "ImageId" : "ami-0ff8a91507f77f867"  
  }  
}
```

# CFT Anatomy

## Parameters – basics

- Enables us to input custom values
- Maximum 200 parameters in a template
- Alphanumeric and unique name
- Parameter Types
  - String
  - Number
  - List of number
  - Comma delimited list e.g) "test", "dev", "prod"
  - AWS-Specific Parameter Types

...

# CFT Anatomy

## Parameters – examples

### Parameters:

#### DBPort:

Default: 3306

Description: TCP/IP port for the database

Type: Number

MinValue: 1150

MaxValue: 65535

#### DBPwd:

NoEcho: true

Description: The database admin account password

Type: String

MinLength: 1

MaxLength: 41

AllowedPattern: `^[a-zA-Z0-9]*$`

# CFT Anatomy

## Parameters – aws specific types

- Predefined parameters
- AWS console **will display you a drop-box** that lists up valid values.
- AWS::EC2::KeyPair::KeyName
- AWS::EC2::Subnet::Id
- AWS::EC2::VPC::Id
- AWS::EC2::Instance::Id
- AWS::EC2::Image::Id
- AWS::EC2::SecurityGroup::Id

...

## Specify stack details

### Stack name

Stack name

Enter a stack name

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

### Parameters

Parameters are defined in your template and allow you to input custom values when you create the stack.

SelectSecurityGroup

Please select a security group.

Select AWS::EC2::SecurityGroup::Id



sg-05e2b431001a91ec1

paimonganyu-skill arm:aws:cloudformation:ap-northeast-2:305992497901:stack/paimonganyu-skill/ef852020-2c48-11ed-b225-0a462d4bf3c8 DeveloperSG

sg-079ad53324891ace3

awseb-e-wpskkpyi9r-stack arm:aws:cloudformation:ap-northeast-2:305992497901:stack/awseb-e-wpskkpyi9r-stack/ae13a070-2c49-11ed-8c68-063f5bf212a8 paim-Paim-t

sg-099243f880e60230c

paimonganyu-skill arm:aws:cloudformation:ap-northeast-2:305992497901:stack/paimonganyu-skill/ef852020-2c48-11ed-b225-0a462d4bf3c8 IKakaoSkillConnectorSG

sg-0ac4a1d67f3d45c73

awseb-e-wpskkpyi9r-stack arm:aws:cloudformation:ap-northeast-2:305992497901:stack/awseb-e-wpskkpyi9r-stack/ae13a070-2c49-11ed-8c68-063f5bf212a8 paim-Paim-t

sg-0eb7bb5ece01db23b

AWS::EC2::SecurityGroup::Id 타입의  
파라미터를 선언한 경우입니다.

보안 그룹 ID 선택을 돕는 드롭 박스가 노출됩니다.

# CFT Anatomy

## Pseudo parameters

- Predefined parameters by AWS CloudFormation
- Use intrinsic function **Ref** to retrieve values
- Popular pseudo parameters
  - `AWS::Region`
  - `AWS::AccountId`
  - `AWS::StackId`
  - `AWS::StackName`

...

# CFT Anatomy

## Pseudo parameters – an example

<JSON>

```
"Outputs": {  
  "StackRegion": {"Value": {"Ref": "AWS::Region"}}  
}
```

<YAML>

```
Outputs:  
  StackRegion:  
    Value: !Ref AWS::Region
```

알아서 정의되는 파라미터 **AWS::Region**의 값을  
Ref 내재함수로 참조해 오는 모습입니다.

...



# Intrinsic Functions(내재 함수)

Several built-in functions used in CFN template

- Fn::Base64
- Fn::Cidr
- Condition Functions
- Fn::FindInMap
- **Fn::GetAtt** – 자원의 특정 속성값을 참조함
- Fn::GetAZs
- Fn::ImportValue
- **Fn::Join** – 문자열 리스트를 delimiter로 조인함
- Fn::Select
- Fn::Split
- **Fn::Sub** – 문자열 플레이스 홀더를 치환함
- Fn::Transform
- **Ref** – 자원의 기본 속성값이나 파라미터의 값을 참조함

# CFT Anatomy

## ④Resources

- Required section
- Defines the AWS resources such as EC2, SNS, S3, ... etc.

Resources:

*Logical ID:*

*Type: Resource type*

*Properties:*

*Set of properties*

...

# CFT Anatomy

## Resources – an S3 bucket

```
AWSTemplateFormatVersion: '2010-09-09'
```

```
Resources:
```

```
  S3Bucket:
```

```
    Type: AWS::S3::Bucket
```

```
Outputs:
```

```
  BucketName:
```

```
    Value: !Ref 'S3Bucket'
```

```
    Description: Name of the sample Amazon S3 bucket.
```

# CFT Anatomy

## Resources – an EC2 instance

```
"Resources" : {  
  "MyEC2Instance" : {  
    "Type" : "AWS::EC2::Instance",  
    "Properties" : {  
      "ImageId" : "ami-012abc124"  
    }  
  }  
}
```

# CFT Anatomy

## Resources – how to describe a resource?

- When you write down a CFT, you **MUST** refer to the AWS CFN documents.
- You should note the required fields for the resources you want to create.
- You may learn a lot from example snippets at below of the documents.
- <https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html>

...

# CFT Anatomy

## ⑤Outputs

- Outputs certain attributes of the resources to outside.  
eg) S3 bucket name, SNS topic name, ... etc.

Outputs:

*Logical ID:*

*Description: Information about the value*

*Value: Value to return*

*Export:*

*Name: Name of resource to export*

# CFT Anatomy

## Outputs – ARN of an SNS topic

```
Outputs:
  PatientTopicArn:
    Description: The ID of the patient topic.
    Value: !Ref PatientTopic
    Export:
      Name: !Sub "${AWS::StackName}-PatientTopic"
```

Stack info | Events | Resources | **Outputs** | Parameters | Template | Change sets

Outputs (1)

Key	Value	Description	Export name
PatientTopicArn	arn:aws:sns:ap-northeast-2:305992497901:patient-topic	The ID of the patient topic.	sns-lambda-PatientTopic

# ... Coding Any CloudFormation Template

1. 아키텍처 뷰를 그린다. 배치할 AWS 자원을 식별한다.
2. (필요하다면 AWS 콘솔에서 아키텍처 배치를 직접 연습해 보자.)
3. 각 자원을 CFT 신택스로 표현하는 법을 구글링한다. ("SQS Queue CloudFormation")
4. [AWS 공식 문서](#)에 당도한다.
5. 자원의 필수 속성 위주로 잘 확인한다. (문서에서 CTRL + F → 'yes', 'conditional' 검색)
6. 예제 템플릿을 꼭 참고한다. 복사하여 템플릿 작성에 활용한다.

정확한 정보 소스를 일관적으로 활용하시면 리서치 속도를 높일 수 있습니다. 📖  
→ 공식 문서, 공식 블로그  
→ 그 시스템을 만드는 사람들과 질답하기 @깃허브, @이슈트래커

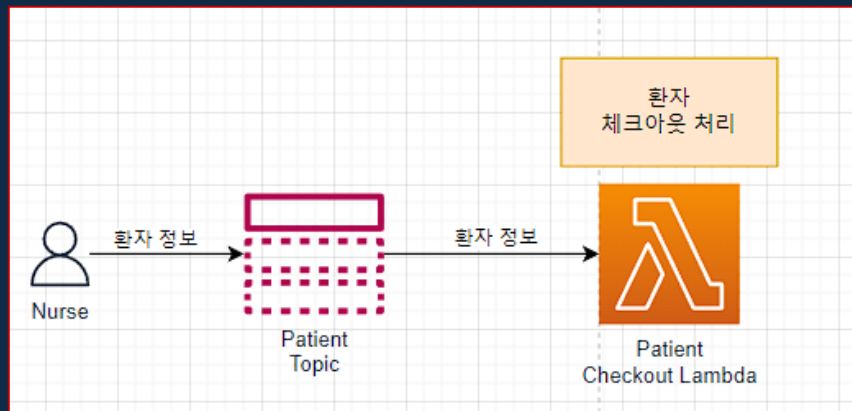


# CloudFormation Demo②

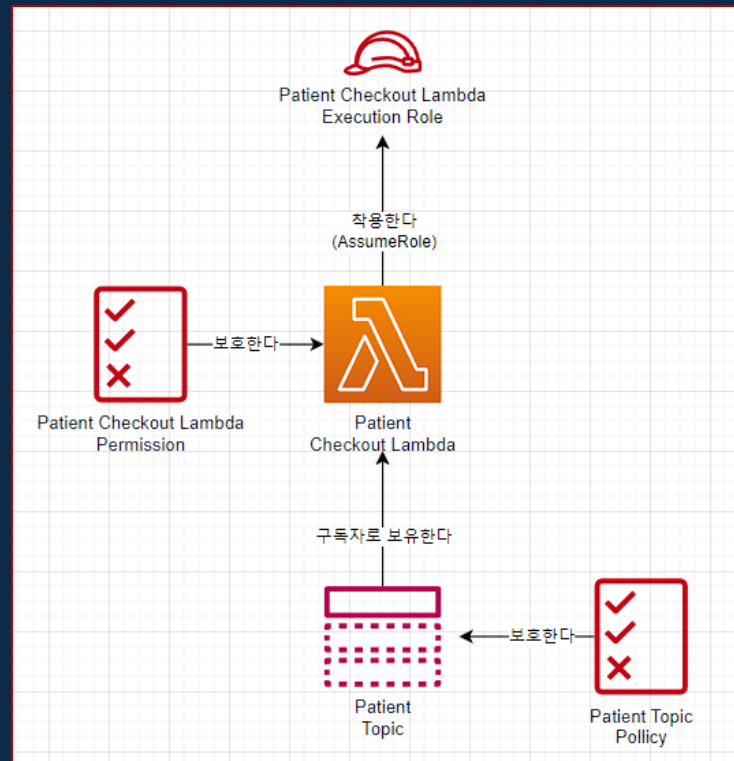
## Subscribing a lambda to an SNS topic

1. Review the reference architecture views.

### 〈Use case〉



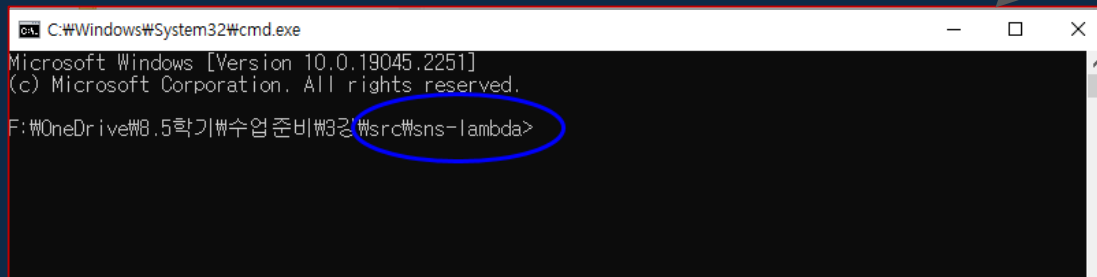
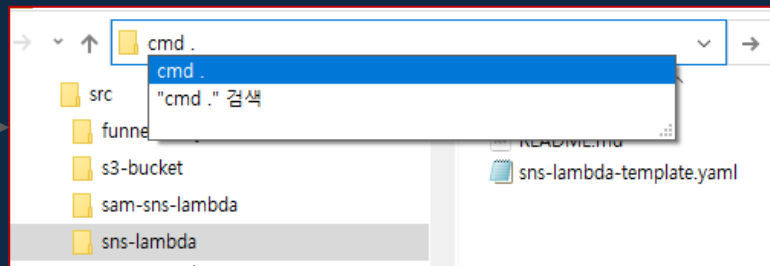
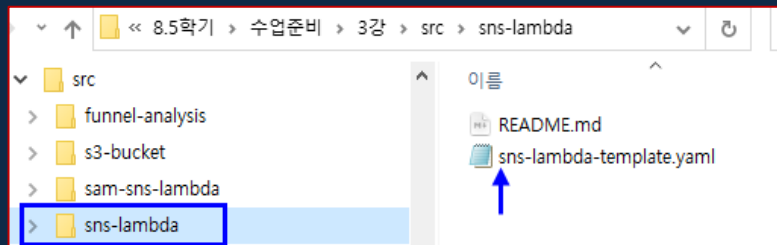
### 〈Deployment & Knowing Responsibilities〉



# CloudFormation Demo②

## Subscribing a lambda to an SNS topic

2. Download the template code @<https://github.com/binchoo/2022-dms-iac/tree/master/src/sns-lambda>
3. Open your cmd or bash at the directory where the `sns-lambda-template.yaml` file is visible.



# CloudFormation Demo②

## Subscribing a lambda to an SNS topic

4. Submit the `sns-lambda-template.yaml` via your AWS CLI.
5. Send a test message to the SNS topic that has been created.

```
aws cloudformation create-stack \  
--stack-name sns-lambda \  
--template-body file://sns-lambda-template.yaml \  
--capabilities CAPABILITY_IAM
```

```
aws sns list-topics  
aws sns publish --topic-arn <topic-arn> \  
--message {"name":"binchoo"}
```

```
{  
  "StackId": "arn:aws:cloudformation:ap-northeast-2:305992497901:stack/sns-lambda  
/31053a20-6ae3-11ed-8dab-0655a5945d70"  
}
```

```
F:\OneDrive\8.5학기\수업 준비\3강\src\sns-lambda>aws sns list-topics  
{  
  "Topics": [  
    {  
      "TopicArn": "arn:aws:sns:ap-northeast-2:305992497901:PaimonganyuReliabi  
lity"  
    },  
    {  
      "TopicArn": "arn:aws:sns:ap-northeast-2:305992497901:paimonganyu-UserHo  
yopassTopic-1013K8RA9Z1ZJ"  
    },  
    {  
      "TopicArn": "arn:aws:sns:ap-northeast-2:305992497901:patient-topic"  
    }  
  ]  
}
```

```
F:\OneDrive\8.5학기\수업 준비\3강\src\sns-lambda>aws sns publish --topic-arn arn:aws  
:sns:ap-northeast-2:305992497901:patient-topic --message {"name":"binchoo"}  
{  
  "MessageId": "84b99fd8-8a8f-5ddf-8263-c42f14eadcbb"  
}
```


# CloudFormation Demo②


## Subscribing a lambda to an SNS topic

6. Verify that `PatientCheckoutFunction` is responding to the messages and generating patient checkout logs.

sns-lambda-PatientCheckoutFunction-vqKe1EF2S4Uf Throttle

▼ Function overview [Info](#)

 sns-lambda-PatientCheckoutFunction-vqKe1EF2S4Uf

 Layers (0)

[+ Add trigger](#)

[+ Add destination](#)

Description

-

Last modified

1 hour ago

Function ARN

arn:aws:lambda:us-east-1:123456789012:function:sns-lambda-PatientCheckoutFunction-vqKe1EF2S4Uf

Application

sns-lambda

Function URL [Info](#)

-

Code Test **Monitor** Configuration Aliases Versions

Metrics **Logs** Traces

[View logs in CloudWatch](#) [View X-Ray traces in ServiceLens](#)

CloudWatch > Log groups > /aws/lambda/sns-lambda-PatientCheckoutFunction-vqKe1EF2S4Uf > 2022/11/22/[\$LATEST]0a1cb45ac34c436ba2d2d40159f4c19f

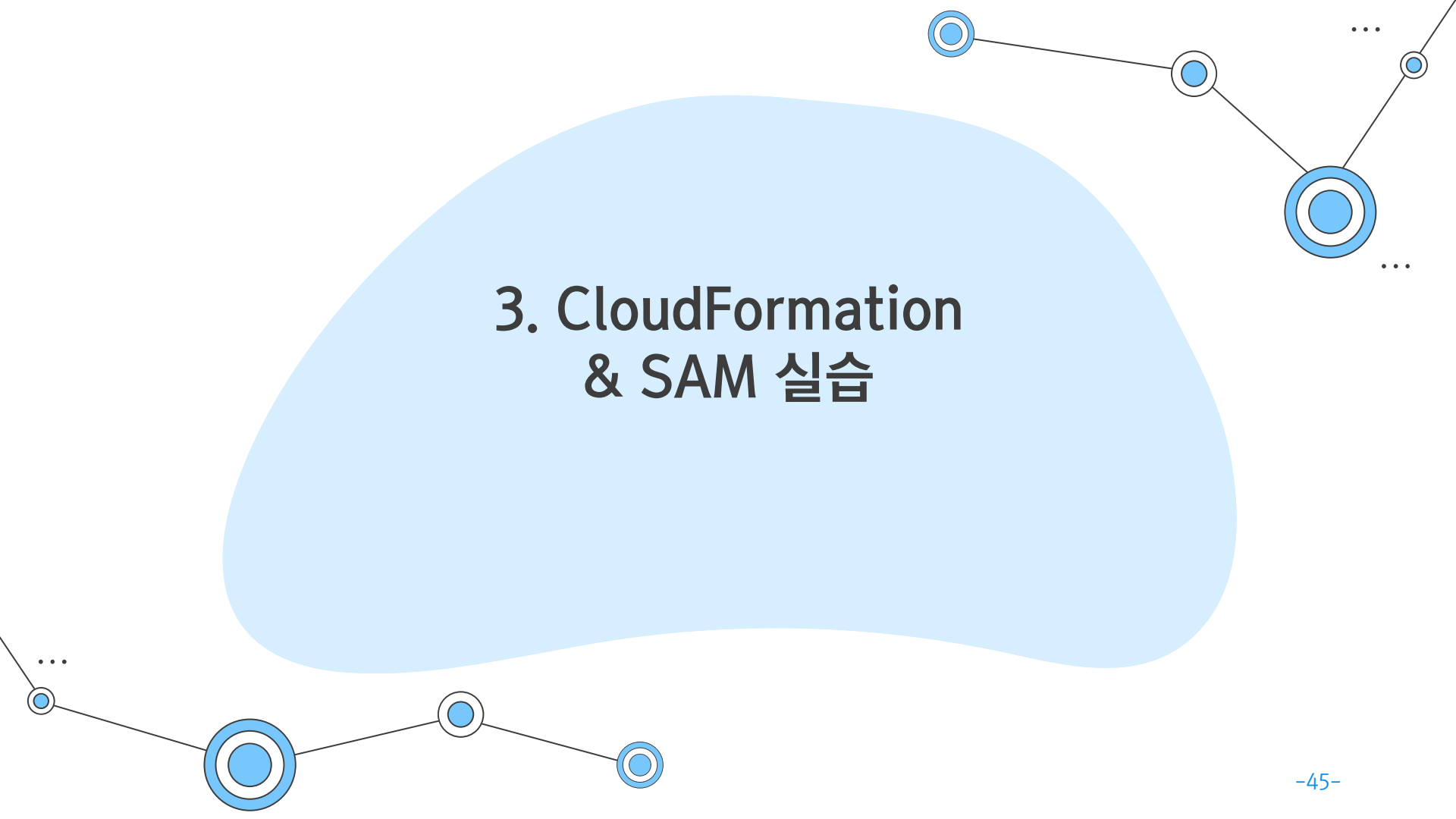
**Log events**

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

[Refresh](#) [Actions](#) [Create metric filter](#)

[Clear](#) [1m](#) [30m](#) [1h](#) [12h](#) [Custom](#)

▶	Timestamp	Message
		No older events at this moment. <a href="#">Retry</a>
▶	2022-11-22T16:10:25.692+09:00	START RequestId: 4c2d000c-cb83-4d04-b5c8-bf60ad5cef62 Version: \$LATEST
▶	2022-11-22T16:10:25.692+09:00	[{"name": "binchoo", "status": "checkout"}]
▶	2022-11-22T16:10:25.693+09:00	END RequestId: 4c2d000c-cb83-4d04-b5c8-bf60ad5cef62
▶	2022-11-22T16:10:25.693+09:00	REPORT RequestId: 4c2d000c-cb83-4d04-b5c8-bf60ad5cef62 Duration: 1.28 ms Billed Duration: 1.28 ms
		No newer events at this moment. <a href="#">Auto retry paused</a> . <a href="#">Resume</a>



### 3. CloudFormation & SAM 실습

# Sample Template(2) – SNS & Lambda

```
AWSTemplateFormatVersion: '2010-09-09'
Description: >
  sns-lambda-template

Example CloudFormation template to subscribe a lambda to an SNS topic.
Resources:
  PatientTopic:
    Type: AWS::SNS::Topic
    Properties:
      DisplayName: patient-topic
      TopicName: patient-topic
      Subscription:
        - Protocol: lambda
          Endpoint: !GetAtt PatientCheckoutFunction.Arn

  PatientTopicPolicy:
    Type: AWS::SNS::TopicPolicy
    Properties:
      Topics:
        - !Ref PatientTopic
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Action: 'sns:Publish'
            Resource: !Ref PatientTopic
            Principal:
              AWS: '*'
            Condition:
              ArnLike:
                AWS:SourceArn: !Sub 'arn:aws:*:*:${AWS::AccountId}:*'

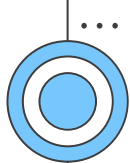
  PatientCheckoutFunction:
    Type: AWS::Lambda::Function
    Properties:
      Runtime: python3.8
      Handler: index.handler
      Code:
        ZipFile: |
          import json
          def handler(event, context):
            patients = [{'name': json.loads(record['Sns']['Message'])['name'], 'st
            print(patients)
      Role: !GetAtt PatientCheckoutFunctionExecutionRole.Arn

  PatientCheckoutFunctionInvokePermission:
    Type: AWS::Lambda::Permission
    Properties:
      Action: 'lambda:InvokeFunction'
      FunctionName: !Ref PatientCheckoutFunction
      Principal: sns.amazonaws.com

  PatientCheckoutFunctionExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Action: '*'
            Resource: '*'
```

구독 설정, 보안 설정, ...  
너무 상황하다!

```
Path: "/"
Policies:
  - PolicyName: root
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Action: '*'
          Resource: '*'
```



# AWS SAM



## Serverless Application Model

- Framework for building and deploying serverless apps
- Resources are configurable with YAML using simple syntax
  - Lambda functions
  - DynamoDB tables
  - API Gateway
  - Cognito user pools
- SAM helps you to test Lambda, API Gateway, DynamoDB locally
  - [sam local invoke](#) ...

# AWS SAM 템플릿

## Transform 영역

- AWS SAM 매크로 이용을 지시한다.
- `AWS::Serverless-2016-10-31`

## AWS::Serverless::Function

- 람다를 나타내는 SAM의 자원 타입
- 서버리스 자원을 짧은 선택스로 명세할 수 있음.

## Events 속성

- 람다와 다양한 이벤트 소스 연동을 단순한 스니펫으로 명세

24줄만으로 Lambda와 SNS 토픽을 만들고 연동시켰습니다.

```
1  AWSTemplateFormatVersion: '2010-09-09'
2  Transform: AWS::Serverless-2016-10-31
3  Description: >
4    sam-sns-lambda-template
5
6  Example AWS SAM template to subscribe a lambda to an SNS topic.
7  Resources:
8    SamPatientTopic:
9      Type: AWS::SNS::Topic
10     Properties:
11       DisplayName: sam-patient-topic
12       TopicName: sam-patient-topic
13
14    SamPatientCheckoutFunction:
15      Type: AWS::Serverless::Function
16     Properties:
17       Runtime: python3.7
18       CodeUri: .
19       Handler: index.handler
20     Events:
21       NewPatient:
22         Type: SNS
23         Properties:
24           Topic: !Ref SamPatientTopic
```



# AWS SAM Flow

## SAM을 통한 람다 소스코드 빌드 및 인프라 배포 흐름

### ① sam build

람다 소스코드와 CFN 템플릿을 빌드



sam build



언어 별  
빌드도구  
(Gradle, Maven)



빌드 된 소스코드



저수준 템플릿

SAM Build 산출물

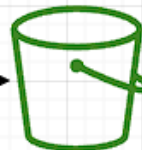
### ② sam deploy

빌드 산출물과 템플릿을 업로드하고, CFN 스택을 생성한다.  
sam package는 sam deploy가 투명하게 수행한다.



sam package

업로드 된다



SAM 전용  
버킷



sam deploy

배치된다



CloudFormation



Stack

-49-



핸들러  
소스코드

참조한다  
(CodeUri와 Handler 속성)



SAM 템플릿

코드 베이스

## AWS Hands-On (3)

- Subscribing a lambda to an SNS topic -

### Covered Services

AWS CloudFormation

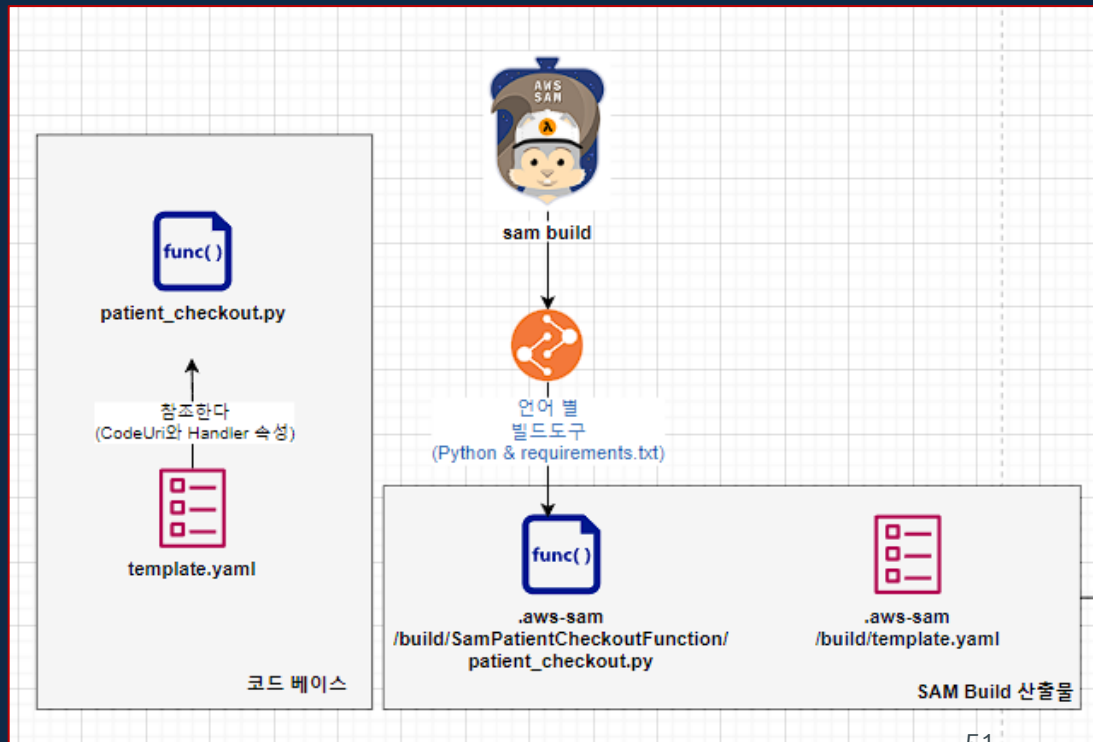
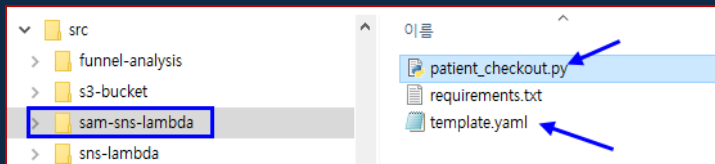
AWS SAM

Amazon SNS

# Reference Architecture View

(예상 소요 시간: 15분)

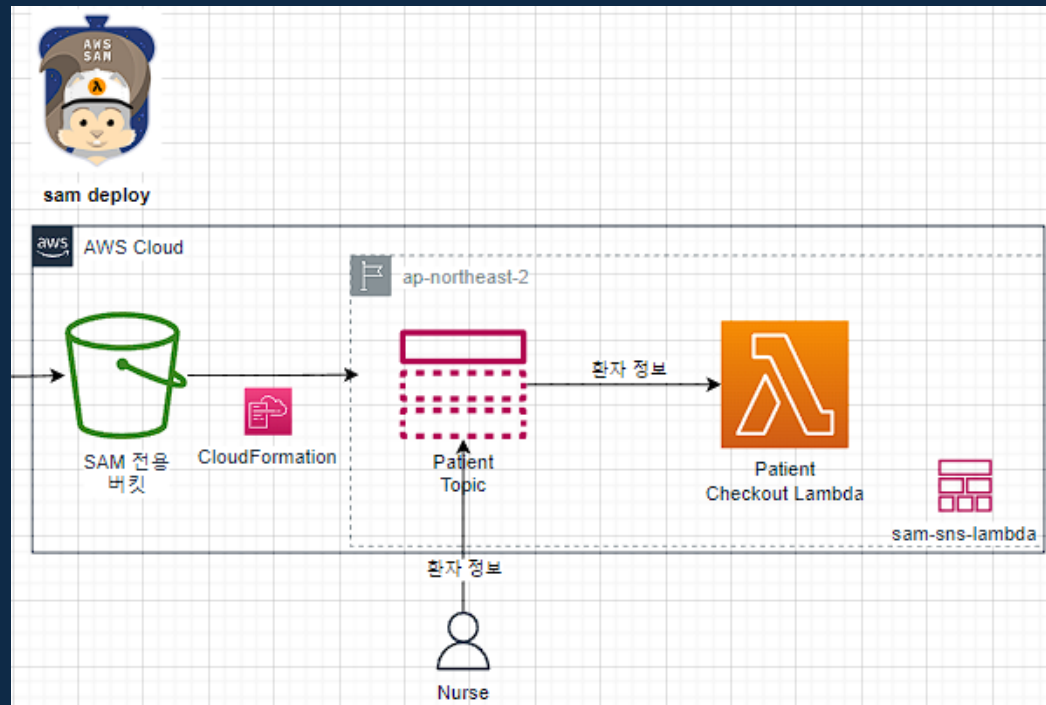
## 로컬 환경



# Reference Architecture View

(예상 소요 시간: 15분)

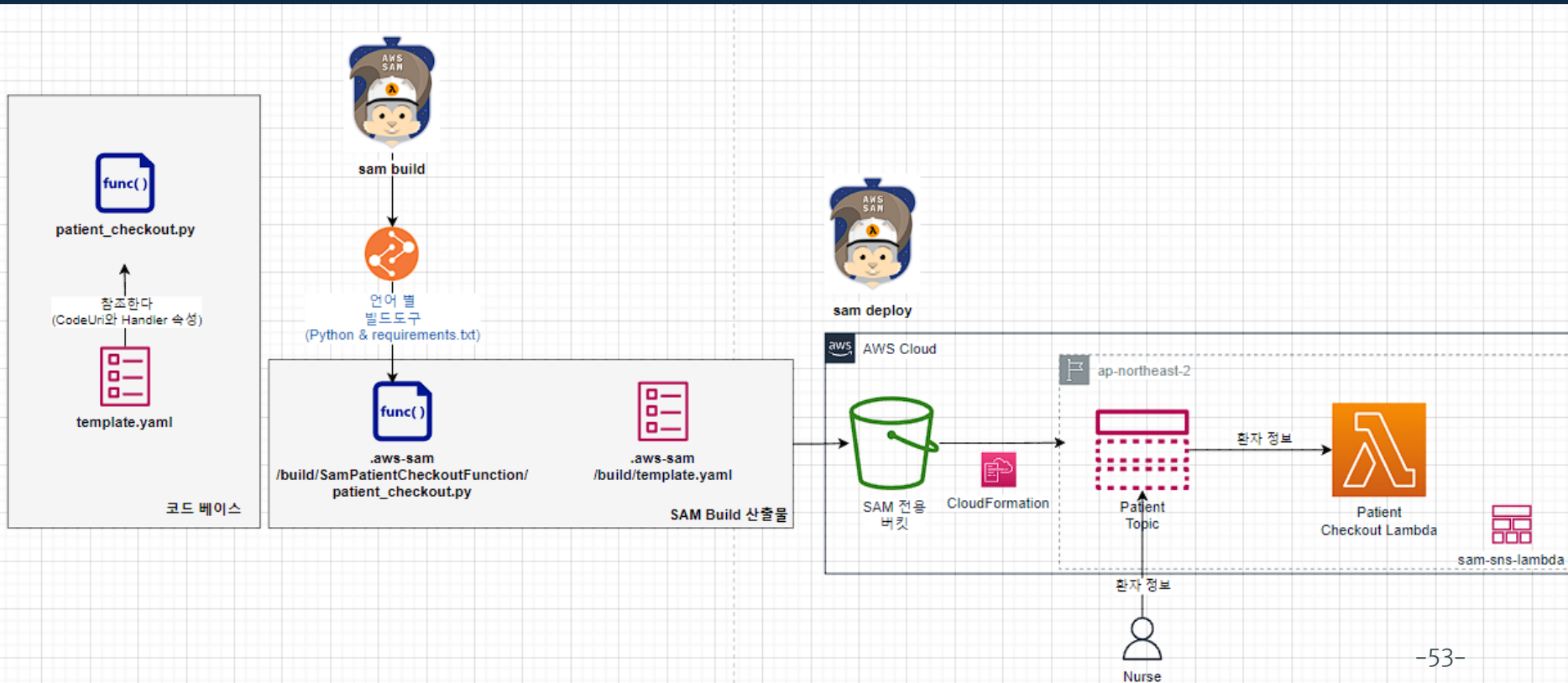
## SAM 배포 이후 AWS 클라우드 환경



# Reference Architecture View

(예상 소요 시간: 15분)

## 전체 뷰



# SAM Template

```
1  AWSTemplateFormatVersion: '2010-09-09'
2  Transform: AWS::Serverless-2016-10-31
3  Description: 'sam-sns-lambda-template'
4
5  Example AWS SAM template to subscribe a lambda to an SNS topic.
6
7  '
8  Resources:
9    SamPatientTopic:
10     Type: AWS::SNS::Topic
11     Properties:
12       DisplayName: sam-patient-topic
13       TopicName: sam-patient-topic
14    SamPatientCheckoutFunction:
15     Type: AWS::Serverless::Function
16     Properties:
17       Runtime: python3.7 # 자기 환경의 Python 버전을 명시하세요 (python3.7, python3.8, python3.9)
18       CodeUri: SamPatientCheckoutFunction
19       Handler: patient_checkout.handler
20     Events:
21       NewPatient:
22         Type: SNS
23         Properties:
24           Topic:
25             Ref: SamPatientTopic
26     Metadata:
27       SamResourceId: SamPatientCheckoutFunction
```

# CloudFormation Demo③

## Subscribing a lambda to an SNS topic with SAM

1. Download the source code @<https://github.com/binchoo/2022-dms-iac/tree/master/src/sam-sns-lambda>
2. Open your cmd or bash.
3. Your python version must be the same with the `SamPatientCheckoutFunction`'s runtime.

```
(base) F:\OneDrive\8.5학기\수업 준비\강\src\sam-sns-lambda>python --version  
Python 3.7.4
```

```
14 SamPatientCheckoutFunction:  
15   Type: AWS::Serverless::Function  
16   Properties:  
17     Runtime: python3.7 # 자기 환경의 Python 버전을 명시하세요 (python3.7, python3.8, python3.9  
18     CodeUri: SamPatientCheckoutFunction  
19     Handler: patient_checkout.handler  
20     Events:  
21       NewPatient:  
22         Type: SNS  
23         Properties:  
24           Topic:  
25             Ref: SamPatientTopic
```

# CloudFormation Demo③

## Subscribing a lambda to an SNS topic with SAM

4. Do `sam build`

5. A new directory `.aws-sam` should have been created and contain artifacts like lambda codes, sam templates, ...etc.

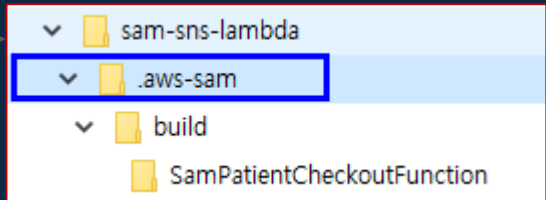
```
(base) F:\OneDrive\8.5학기\수업 준비\강\src\sam-sns-lambda>sam build
Building codeuri: F:\OneDrive\8.5학기\수업 준비\강\src\sam-sns-lambda runtime: python3.7
Architecture: x86_64 functions: SamPatientCheckoutFunction
Running PythonPipBuilder:ResolveDependencies
Running PythonPipBuilder:CopySource

Build Succeeded

Built Artifacts  : .aws-sam\build
Built Template   : .aws-sam\build\template.yaml

Commands you can use next
=====
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
[*] Deploy: sam deploy --guided
```

It's new!!





# CloudFormation Demo③

## Subscribing a lambda to an SNS topic with SAM

6. Do `sam deploy --guided`
7. Input some configuration values for the CFN stack.
8. Sam Deploy will display you the change-set.
9. Sam Deploy will deploy the resources into the CFN stack. You may see the realtime logs of the deployment.

```
(base) F:\OneDrive\3.5학기\수업 준비\3강\src\sam-sns-lambda>sam deploy --guided
```

### Configuring SAM deploy

=====

```
Looking for config file [samconfig.toml] : Not found
```

```
Setting default arguments for 'sam deploy'
```

=====

```
Stack Name [sam-app]: sam-sns-lambda
```

```
AWS Region [ap-northeast-2]: ap-northeast-2
```

```
#Shows you resources changes to be deployed and require a 'Y' to initiate deploy
```

```
Confirm changes before deploy [y/N]: 엔터
```

```
#SAM needs permission to be able to create roles to connect to the resources in your template
```

```
Allow SAM CLI IAM role creation [Y/n]: 엔터
```

```
#Preserves the state of previously provisioned resources when an operation fails
```

```
Disable rollback [y/N]: 엔터
```

```
Save arguments to configuration file [Y/n]: 엔터
```

```
SAM configuration file [samconfig.toml]: 엔터
```

```
SAM configuration environment [default]: 엔터
```

스택 이름을 뭐로 할까요? sam-sns-lambda

어느 리전에 배포할까요? 서울이요

ChangeSet 검수할래요? 아뇨

IAM 롤 생성을 허용할래요? 네

배포 중 오류시 롤백합니까? 네

지금 설정하신 값들 로컬에 저장할래요? 네

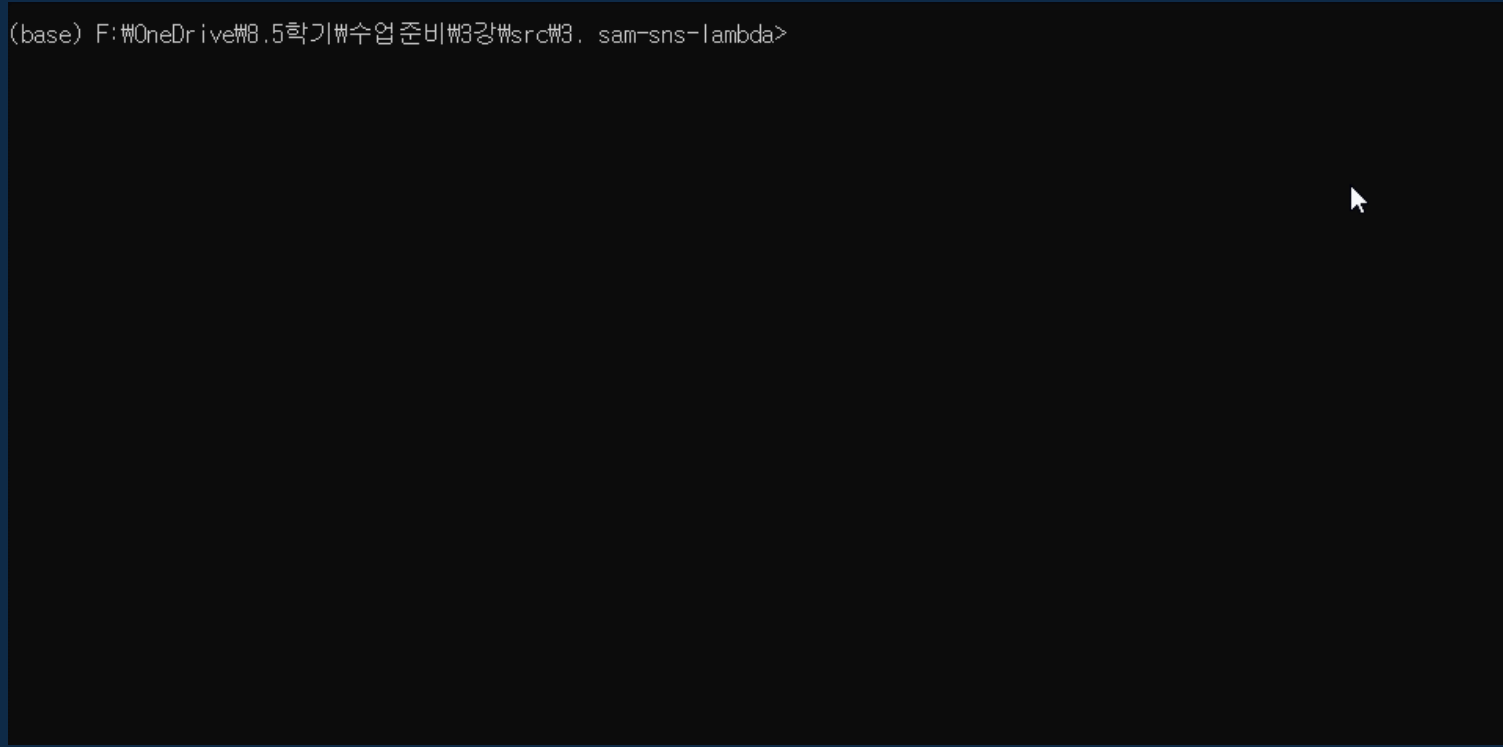
어느 파일에 저장하나요? samconfig.toml

지금 설정하신 값들을 어떤 묶음으로 관리할래요? default

# CloudFormation Demo③

## Sam deploy 과정 GIF 이미지

(base) F:\OneDrive\8.5학기\수업 준비\강\src\3. sam-sns-lambda>



# CloudFormation Demo③

## Change set 출력 예

Waiting for changeset to be created..  
CloudFormation stack changeset

Operation	LogicalResourceId	ResourceType	Replacement
+ Add	SamPatientCheckoutFunctionNewPatientPermission	AWS::Lambda::Permission	N/A
+ Add	SamPatientCheckoutFunctionNewPatient	AWS::SNS::Subscription	N/A
+ Add	SamPatientCheckoutFunctionRole	AWS::IAM::Role	N/A
+ Add	SamPatientCheckoutFunction	AWS::Lambda::Function	N/A
+ Add	SamPatientTopic	AWS::SNS::Topic	N/A

Changeset created successfully. arn:aws:cloudformation:ap-northeast-2:305992497901:changeSet/samcli-deploy1669115286/4c5facef-2294-4cfe-8c70-5f7b6348202a

# CloudFormation Demo③

## 스택 이벤트 출력 예

CloudFormation events from stack operations (refresh every 0.5 seconds)

ResourceStatus	ResourceType	LogicalResourceId	ResourceStatusReason
CREATE_IN_PROGRESS	AWS::IAM::Role	SamPatientCheckoutFunctionRole	-
CREATE_IN_PROGRESS	AWS::SNS::Topic	SamPatientTopic	-
CREATE_IN_PROGRESS	AWS::IAM::Role	SamPatientCheckoutFunctionRole	Resource creation Initiated
CREATE_IN_PROGRESS	AWS::SNS::Topic	SamPatientTopic	Resource creation Initiated
CREATE_COMPLETE	AWS::SNS::Topic	SamPatientTopic	-
CREATE_COMPLETE	AWS::IAM::Role	SamPatientCheckoutFunctionRole	-
CREATE_IN_PROGRESS	AWS::Lambda::Function	SamPatientCheckoutFunction	-
CREATE_IN_PROGRESS	AWS::Lambda::Function	SamPatientCheckoutFunction	Resource creation Initiated
CREATE_COMPLETE	AWS::Lambda::Function	SamPatientCheckoutFunction	-
CREATE_IN_PROGRESS	AWS::Lambda::Permission	SamPatientCheckoutFunctionNewPatientPermission	-
CREATE_IN_PROGRESS	AWS::SNS::Subscription	SamPatientCheckoutFunctionNewPatient	-
CREATE_IN_PROGRESS	AWS::Lambda::Permission	SamPatientCheckoutFunctionNewPatientPermission	Resource creation Initiated
CREATE_IN_PROGRESS	AWS::SNS::Subscription	SamPatientCheckoutFunctionNewPatient	Resource creation Initiated
CREATE_COMPLETE	AWS::SNS::Subscription	SamPatientCheckoutFunctionNewPatient	-
CREATE_COMPLETE	AWS::Lambda::Permission	SamPatientCheckoutFunctionNewPatientPermission	-
CREATE_COMPLETE	AWS::CloudFormation::Stack	sam-sns-lambda	-

Successfully created/updated stack - sam-sns-lambda in ap-northeast-2

# CloudFormation Demo③

## Subscribing a lambda to an SNS topic with SAM

10. Validate your CFN console has the `sam-sns-lambda` stack.

Stacks (6)

Delete

Update

Stack actions ▾

Create stack ▾

🔍 Filter by stack name

View nested

Active ▾

< 1 >

	Stack name	Status	Created time ▾	Description
<div><div></div></div>	sam-sns-lambda	<div><div></div>CREATE_COMPLETE</div>	2022-11-22 20:08:06 UTC+0900	sam-sns-lambda-template Example AWS SAM template to subscribe a lambda to an SNS topic.

# CloudFormation Demo③

## Subscribing a lambda to an SNS topic with SAM

11. Send a test message to the SNS topic that has been created within your CFN stack.

```
aws sns list-topics
aws sns publish --topic-arn <topic-arn> \
--message {"name":"binchoo"}
```

```
(base) F:\OneDrive\8.5학기\수업 준비\8강\src\sam-sns-lambda>aws sns list-topics
{
  "Topics": [
    {
      "TopicArn": "arn:aws:sns:ap-northeast-2:305992497901:PaimonganyuReliability"
    },
    {
      "TopicArn": "arn:aws:sns:ap-northeast-2:305992497901:paimonganyu-UserHoyopassTopic-1013K8PA9Z1ZJ"
    },
    {
      "TopicArn": "arn:aws:sns:ap-northeast-2:305992497901:patient-topic"
    },
    {
      "TopicArn": "arn:aws:sns:ap-northeast-2:305992497901:sam-patient-topic"
    }
  ]
}
```

```
F:\OneDrive\8.5학기\수업 준비\8강\src\8. sam-sns-lambda>aws sns publish --topic-arn arn:aws:sns:ap-northeast-2:305992497901:sam-patient-topic --message {"name":"binchoo"}
{
  "MessageId": "ee7b5199-a1cc-5eb4-8f69-0a6f12bd2630"
}
```

# CloudFormation Demo③

## Subscribing a lambda to an SNS topic with SAM

12. Verify that `SamPatientCheckoutFunction` is responding to the messages and generating patient checkout logs.

sam-sns-lambda-SamPatientCheckoutFunction-9Su0m2SLuMcV

This function belongs to an application. [Click here](#) to manage it.

▼ Function overview [Info](#)

SamPatientCheckoutFunction-9Su0m2SLuMcV

Layers (0)

SNS

+ Add trigger

+ Add destination

Description

Last modified: 14 minutes ago

Function ARN: arn:aws:lambda:ap-da-SamPatientChecko

Application: sam-sns-lambda

Function URL: [Info](#)

Code Test **Monitor** Configuration Aliases Versions

Metrics **Logs** Traces

[View logs in CloudWatch](#) [View X-Ray traces](#)

Timestamp	Message
	No older events at this moment. <a href="#">Retry</a>
2022-11-22T20:23:29.464+09:00	START RequestId: 1ba2d49c-5014-4007-95b9-e16ab90cd86c Version: \$LATEST
2022-11-22T20:23:29.465+09:00	<code>{{'name': 'binchoo', 'status': 'checkout'}}</code>
2022-11-22T20:23:29.467+09:00	END RequestId: 1ba2d49c-5014-4007-95b9-e16ab90cd86c
2022-11-22T20:23:29.467+09:00	REPORT RequestId: 1ba2d49c-5014-4007-95b9-e16ab90cd86c Duration: 2.56 ms B
	No newer events at this moment. <a href="#">Auto retry paused</a> . <a href="#">Resume</a>

로그 스트림> 환자 checkout 상태 확인

Lambda 콘솔> SamPatientCheckoutFunction> CloudWatch Logs 이동

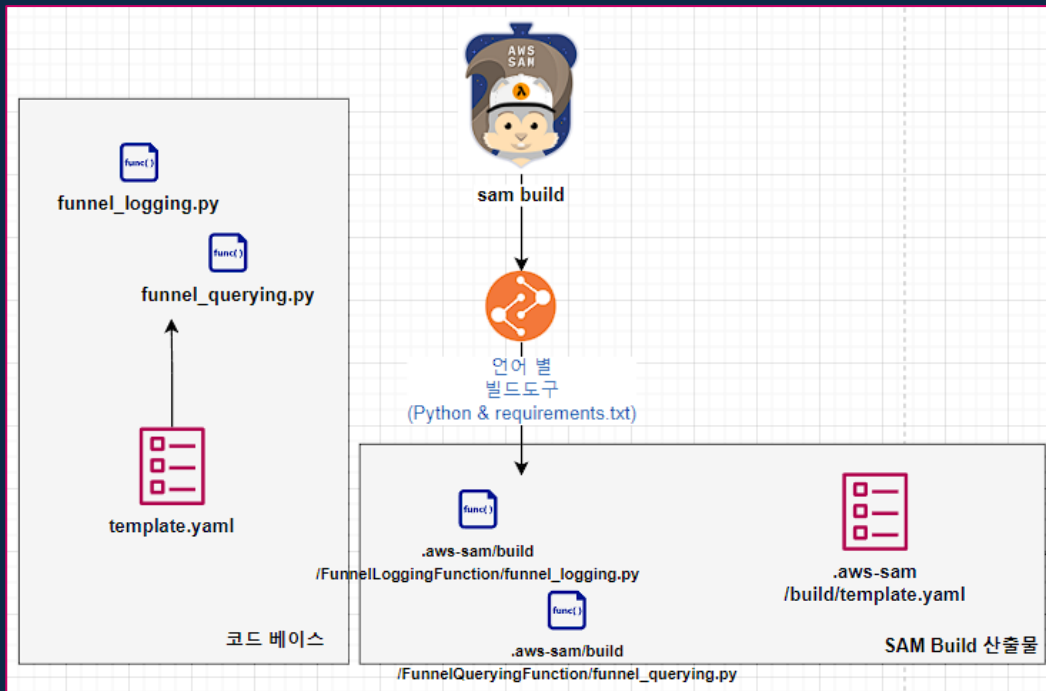
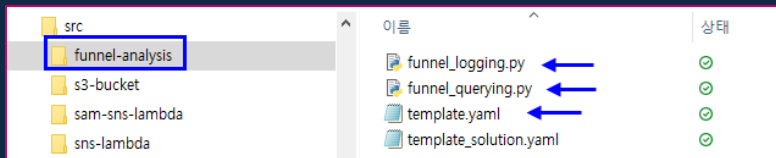
**[DIY]**  
**Building a Funnel Analysis Pipeline  
with AWS SAM**



# Reference Architecture View

(예상 소요 시간: 20분)

## 로컬 환경

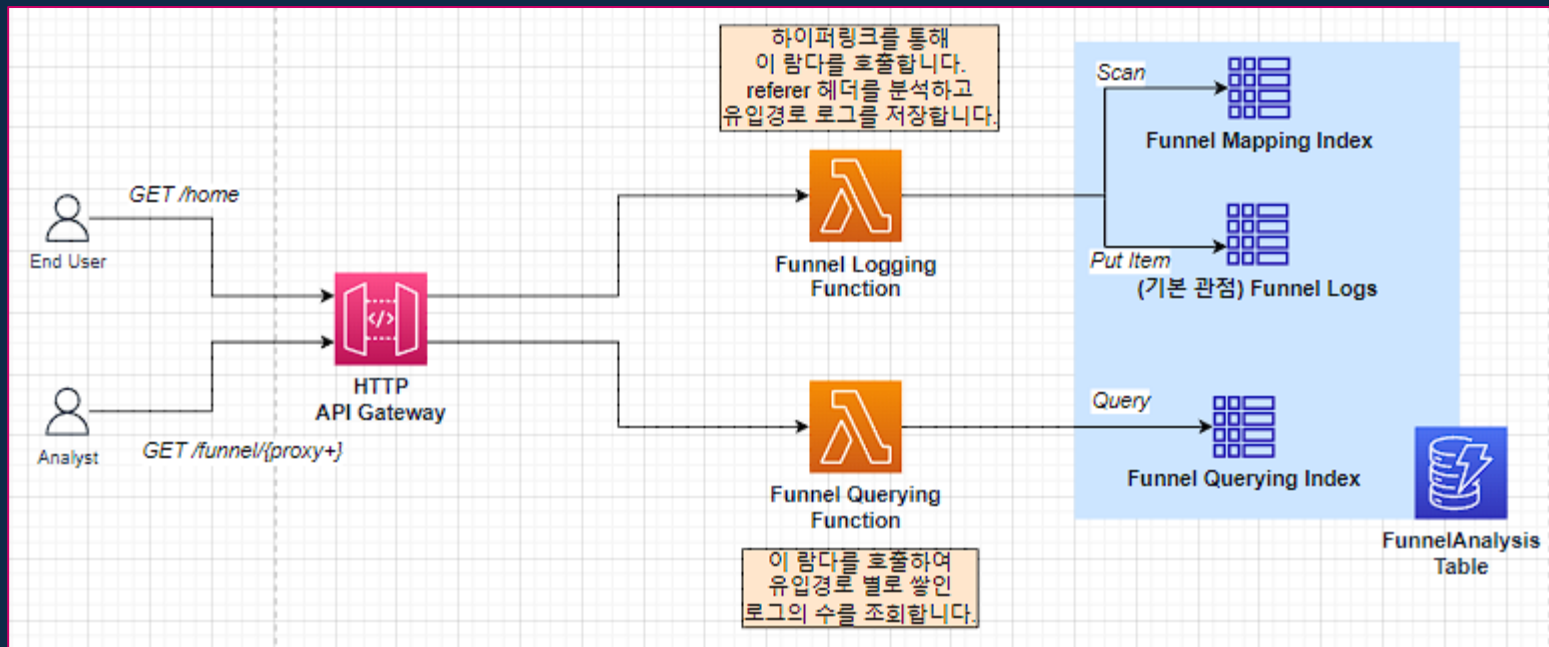


# Reference Architecture View

(예상 소요 시간: 20분)

## SAM 배포 이후 AWS 클라우드 환경

지난 실습 아키텍처와 별반 다르지 않습니다.



# CloudFormation DIY

## Funnel Analysis Pipeline

1. Download the source code from <https://github.com/binchoo/2022-dms-iac/tree/master/src/funnel-analysis>
2. Open your cmd or bash.
3. Inspect whether your python version is the same with the functions' runtime environments.  
(FunnelLoggingFunction, FunnelQueryingFunction)

```
(base) F:\OneDrive\8.5학기\수업 준비\강\src\sam\sns-lambda>python --version  
Python 3.7.4
```

```
FunnelLoggingFunction: 63  
  Type: AWS::Serverless::Function 64  
  Properties: 65  
    CodeUri: . 66  
    Handler: funnel_logging.handler 67  
    Runtime: python3.7 # TODO: 여러분 파이썬 68  
    Policies: 69  
      - DynamoDBCrudPolicy: 70  
        TableName: !Ref FunnelAnalysisTable 71  
  Events: 72  
    GetHomeEvent: 73  
      Type: HttpApi 74  
      Properties: 75  
        ApiId: !Ref FunnelApi 76  
        Method: get 77  
        Path: /home 78  
79  
FunnelQueryingFunction: 63  
  Type: AWS::Serverless::Function 64  
  Properties: 65  
    CodeUri: . 66  
    Handler: funnel_querying.handler 67  
    Runtime: python3.7 # TODO: 여러분 파이썬 68  
    Policies: 69  
      - DynamoDBReadPolicy: 70  
        TableName: !Ref FunnelAnalysisTable 71  
  FunnelApi: 72  
    Type: AWS::Serverless::HttpApi 73  
  FunnelQueryRoute: 74  
    Type: AWS::ApiGatewayV2::Route 75  
    Properties: 76  
      ApiId: !Ref FunnelApi 77  
78  
79
```


# CloudFormation DIY

## Funnel Analysis Pipeline

4. You specify that the `FunnelLoggingFunction` has the `GET /home` endpoint of `FunnelApi` as its trigger. You should fix some lines that are `TODO` annotated.
5. Also, make the `FunnelLoggingFunction` able to access the `FunnelAnalysisTable`.

```
FunnelLoggingFunction:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: .
    Handler: ??? #TODO: 수정
    Runtime: python3.7 # TODO: 여러분 파이썬 환경으로 바꾸세요 (python3.7, python3.8, python3.9)
    Policies:
      - DynamoDBCrudPolicy:
          TableName: ??? #TODO: 수정
    Events:
      GetHomeEvent:
        Type: HttpApi
        Properties:
          ApiId: ??? #TODO: 수정
          Method: ??? #TODO: 수정
          Path: ??? #TODO: 수정
```

Conduct your research yourself!

 [aws sam function doc](#)

 [aws sam function predefined policies](#)

# CloudFormation DIY

## Funnel Analysis Pipeline

6. Do `sam build`

7. Do `sam deploy --guided`. The configuration values are shown in the image.

```
(base) F:\OneDrive\8.5학기\수업 준비\3강\src\funnel-analysis> sam deploy --guided

Configuring SAM deploy
=====

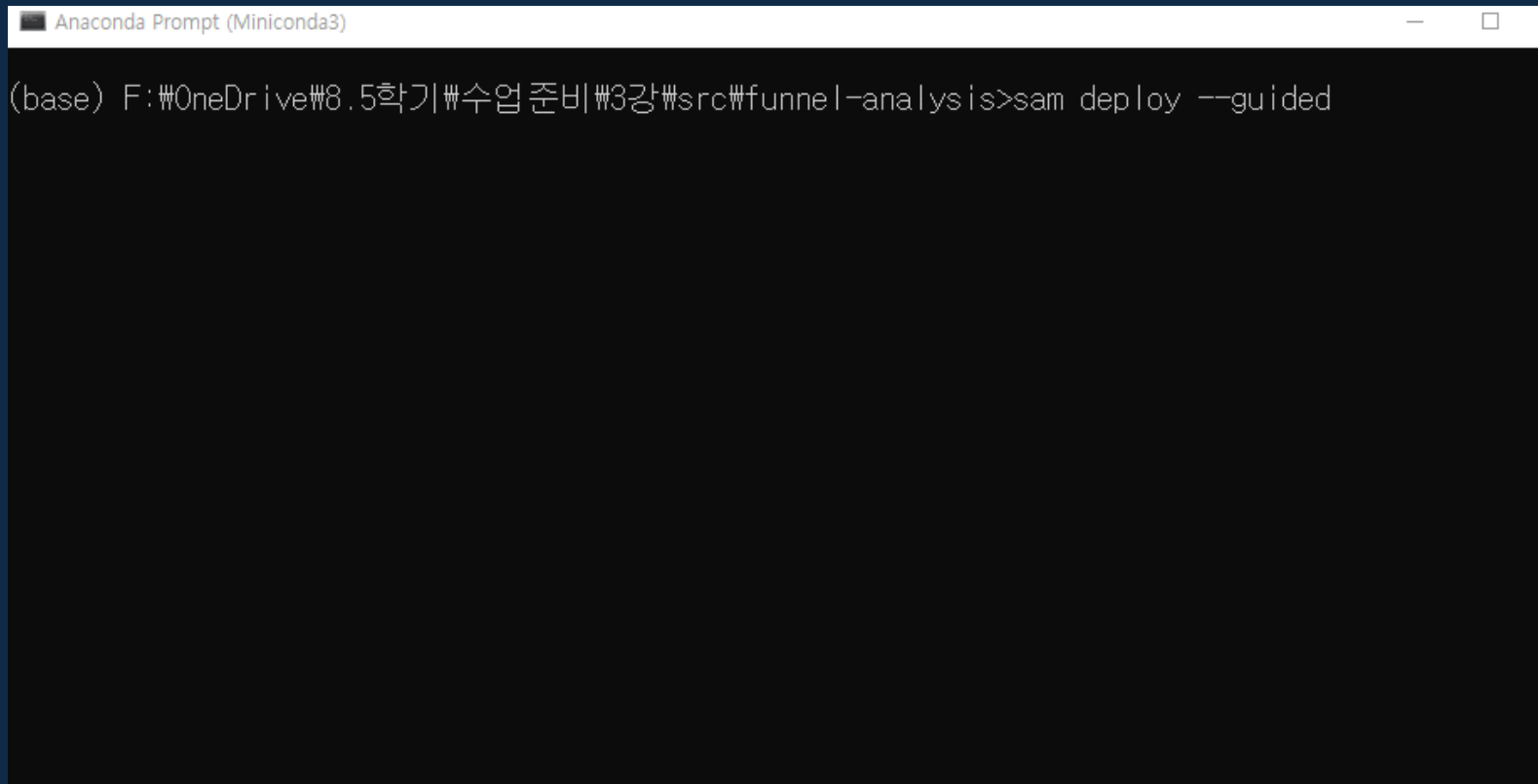
Looking for config file [samconfig.toml] : Not found

Setting default arguments for 'sam deploy'
=====
Stack Name [sam-app]: funnel-analysis
AWS Region [ap-northeast-2]:
#Shows you resources changes to be deployed and require a 'Y' to initiate deploy
Confirm changes before deploy [y/N]:
#SAM needs permission to be able to create roles to connect to the resources in your template

Allow SAM CLI IAM role creation [Y/n]:
#Preserves the state of previously provisioned resources when an operation fails
Disable rollback [y/N]:
FunnelLoggingFunction may not have authorization defined, Is this okay? [y/N]: Y
Save arguments to configuration file [Y/n]:
SAM configuration file [samconfig.toml]:
SAM configuration environment [default]:
```

# CloudFormation Demo③

## Sam deploy 과정 GIF 이미지

A screenshot of an Anaconda Prompt terminal window. The title bar at the top reads "Anaconda Prompt (Miniconda3)". The terminal content shows a command prompt "(base) F:\OneDrive\8.5학기\수업 준비\3강\src\funnel-analysis>" followed by the command "sam deploy --guided". The rest of the terminal area is empty.

```
(base) F:\OneDrive\8.5학기\수업 준비\3강\src\funnel-analysis>sam deploy --guided
```

# CloudFormation DIY

## Funnel Analysis Pipeline

6. Go to AWS CloudFormation Console › Stack: `funnel-analysis` › Outputs

7. Play with the API endpoints.

The screenshot shows the AWS CloudFormation console interface. At the top, the breadcrumb navigation reads 'CloudFormation > Stacks > funnel-analysis'. On the left sidebar, under 'Stacks (5)', the 'funnel-analysis' stack is selected, showing a status of 'CREATE\_COMPLETE'. The main panel displays the 'funnel-analysis' stack details, with the 'Outputs' tab highlighted. Below the tab, there are four outputs listed in a table:

Key	Value	Description
FacebookCountEndpoint	<a href="https://ufafij3qih.execute-api.ap-northeast-2.amazonaws.com/funnel/facebook">https://ufafij3qih.execute-api.ap-northeast-2.amazonaws.com/funnel/facebook</a>	-
HomeEndpoint	<a href="https://ufafij3qih.execute-api.ap-northeast-2.amazonaws.com/home">https://ufafij3qih.execute-api.ap-northeast-2.amazonaws.com/home</a>	-
InstagramCountEndpoint	<a href="https://ufafij3qih.execute-api.ap-northeast-2.amazonaws.com/funnel/instagram">https://ufafij3qih.execute-api.ap-northeast-2.amazonaws.com/funnel/instagram</a>	-
TwitterCountEndpoint	<a href="https://ufafij3qih.execute-api.ap-northeast-2.amazonaws.com/funnel/twitter">https://ufafij3qih.execute-api.ap-northeast-2.amazonaws.com/funnel/twitter</a>	-

**THANK  
YOU**

