

Activity 4

Locate the template in the GitHub repository, save it locally, and open it in VS Code for edits. Add a Mappings section for a RegionImages map, defining AMIs for eu-west-1 and us-east-1 under the ImageId key. Replace the ImageId parameter with Fn::FindInMap, referencing the map, AWS::Region, and ImageId. Add a Metadata section to group and label parameters into Web Server Settings and VPC Settings. Save and test by creating stacks in AWS Ireland and North Virginia, verifying AMI selection and parameter grouping. Delete both stacks to clean up resources after validation.

What we have Done in this Activity:

1. Dynamic Region Mapping:

- Defined a Mappings section to provide region-specific AMIs for EC2 instances and used the Fn::FindInMap function to dynamically retrieve AMIs based on the AWS region.

2. Improved Parameter Management:

- Introduced a Metadata section to organize parameters into logical groups (Web Server Settings and VPC Settings) and applied custom labels to improve user clarity.

3. Validation Across Regions:

- Verified the template's functionality by successfully creating and testing stacks in multiple AWS regions (Ireland and North Virginia) and ensured the mappings worked correctly.

4. Simplified User Experience:

- Removed redundant parameters (like ImageId) to streamline the stack creation process.

Activity

1. Find the template files in our GitHub repository under the same name as the heading for easy access and edits. Find and Save the attached template locally, open it in VS Code for edits.

2. Define a Mappings section between Parameters and Resources. Create a map named RegionImages with top-level keys for eu-west-1 and us-east-1.

```
50
51 Mappings:
52   RegionImages:
53     eu-west-1:
54       ImageId: ami-04bd4a6a67aa8e86e
55     us-east-1:
56       ImageId: ami-03945116ac87ab953
57
```

3. Add a second-level key ImageId with valid AMIs for each region. Update the WebServerInstance resource's ImageId to use the Fn::FindInMap function, referencing the map, the AWS::Region pseudo parameter, and the ImageId key.
4. Remove the existing ImageId parameter from the template and add the highlighted data.

```
148 Properties:
149   InstanceType: !Ref InstanceType
150   SubnetId: !Ref PublicSubnet
151   ImageId: !FindInMap [ RegionImages, !Ref 'AWS::Region', ImageId ]
152   KeyName: !Ref KeyPairName
153   SecurityGroupIds:
```

5. Add a Metadata section for parameter grouping and labeling before Parameters. Define ParameterGroups for Web Server Settings and VPC Settings, assigning relevant parameters to each. Customize labels for specific parameters like SubnetAZ and KeyPairName under ParameterLabels in the metadata.

```
8 Metadata:
9   AWS::CloudFormation::Interface:
10     ParameterGroups:
11       - Label:
12         default: Web Server Settings
13       - Label:
14         default: VPC Settings
15     Parameters:
16       - InstanceType
17       - EbsVolumeSize
18       - KeyPairName
19       - Label:
20         default: VPC Settings
21     Parameters:
22       - VpcCidrBlocks
23       - SubnetAZ
24     ParameterLabels:
25       SubnetAZ:
26         default: Select an Availability Zone for the subnets
27       KeyPairName:
28         default: Select an EC2 key pair
```

6. Save the updated template and test it by creating a stack in AWS Ireland.

7. Verify correct parameter grouping, labels, and stack creation.

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

Web Server Settings

InstanceType
t2.micro

EbsVolumeSize
Volume size in GiB
10

Select an EC2 key pair
key-1

VPC Settings

VpcCidrBlocks
vpc, public subnet, private subnet
10.0.0.0/16, 10.0.1.0/24, 10.0.2.0/24

Select an Availability Zone for the subnets
eu-west-1c

Cancel Previous Next

8. Switch to AWS North Virginia, upload the template, and create another stack.

9. Validate stack functionality and verify the correct AMI is used for each region.

CloudFormation > Stacks > databasestack

Stacks (1)
Filter by stack name

Filter status
Active View nested

Stacks
databasestack
2024-12-19 14:20:11 UTC+0530
CREATE_IN_PROGRESS

databasestack

Stack info Events - updated Resources Outputs Parameters

Events (32)
Search events

Timestamp	Logical ID	Status
2024-12-19 14:20:35 UTC+0530	WebServerSecurityGroup	CREATE_COMPLETE
2024-12-19 14:20:31 UTC+0530	PublicSubnetRouteTableAssoc	CREATE_COMPLETE
2024-12-19 14:20:31 UTC+0530	PrivateSubnetRouteTableAssoc	CREATE_COMPLETE
2024-12-19 14:20:31 UTC+0530	PublicSubnetRouteTableAssoc	CREATE_IN_PROGRESS
2024-12-19 14:20:31	PrivateSubnetRouteTable	CREATE_IN_PROGRESS

United States
N. Virginia us-east-1
Ohio us-east-2
N. California us-west-1
Oregon us-west-2
Asia Pacific
Mumbai ap-south-1
Osaka ap-northeast-3
Seoul ap-northeast-2
Singapore ap-southeast-1
Sydney ap-southeast-2
Tokyo ap-northeast-1
Canada
Central ca-central-1
Europe
Frankfurt eu-central-1
Ireland eu-west-1
London eu-west-2
Paris eu-west-3
Stockholm eu-north-1
Middle East
UAE me-central-1

Create stack

Timeline view - new

Initiated

10. Delete stacks in both regions to clean up resources.

