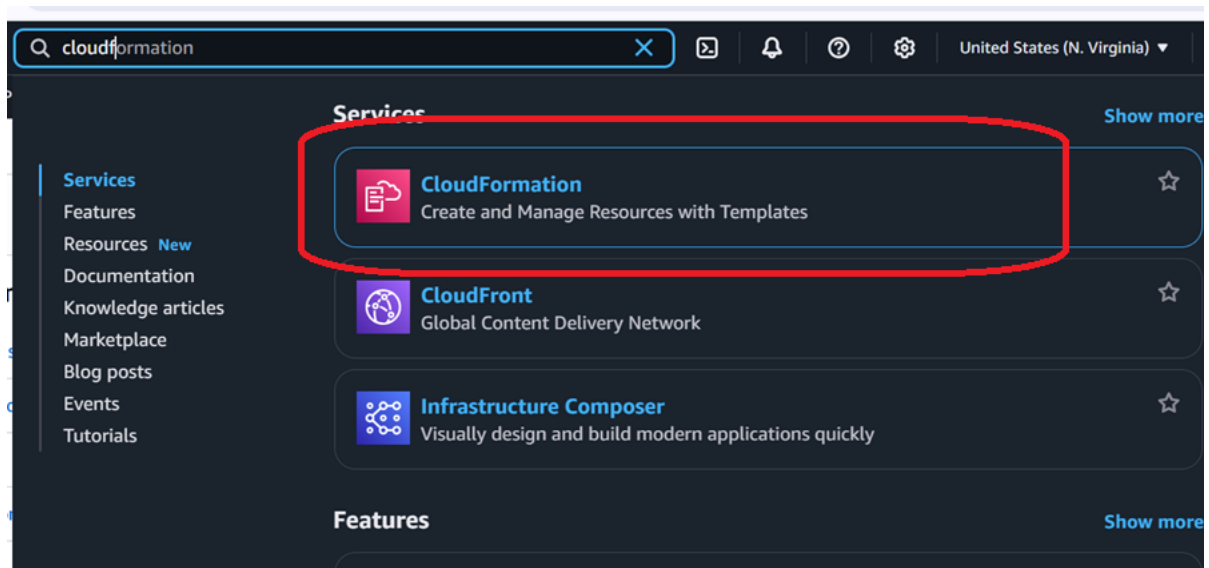
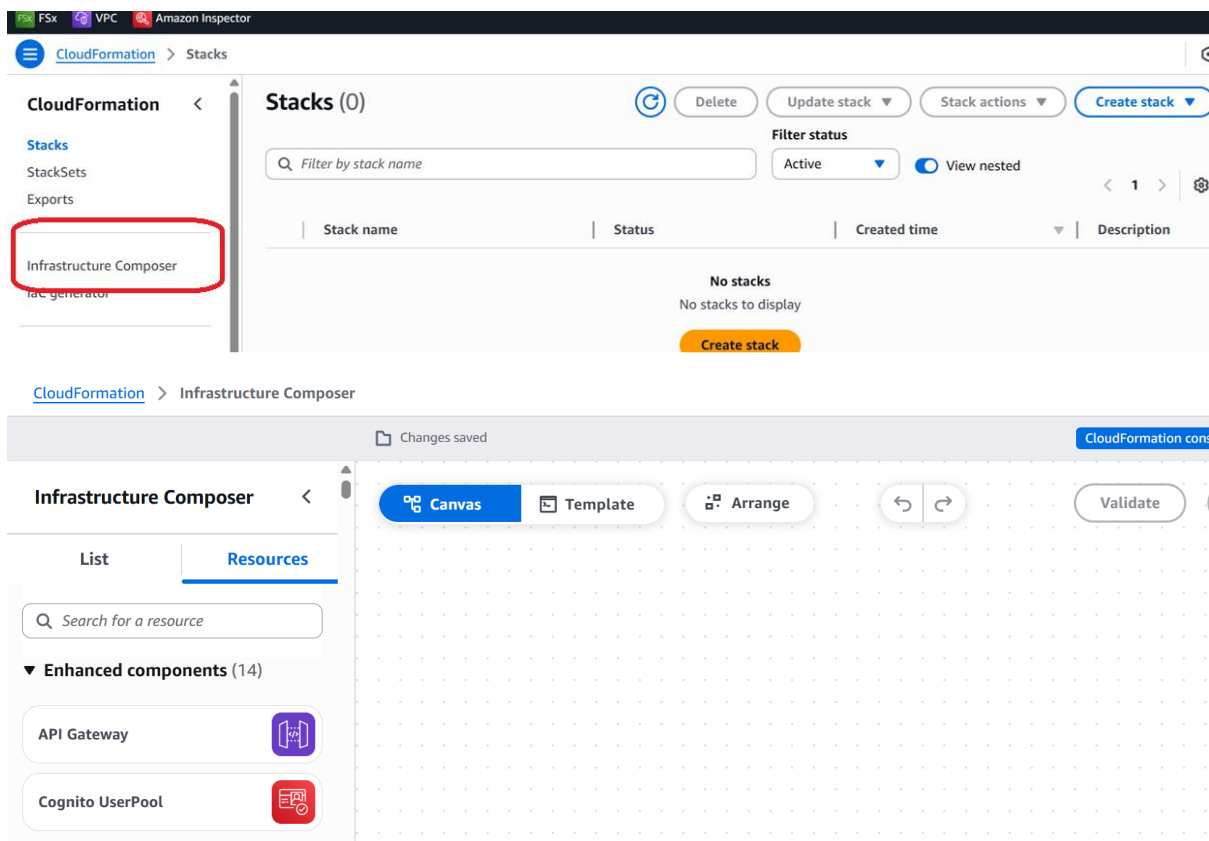


CloudFormation

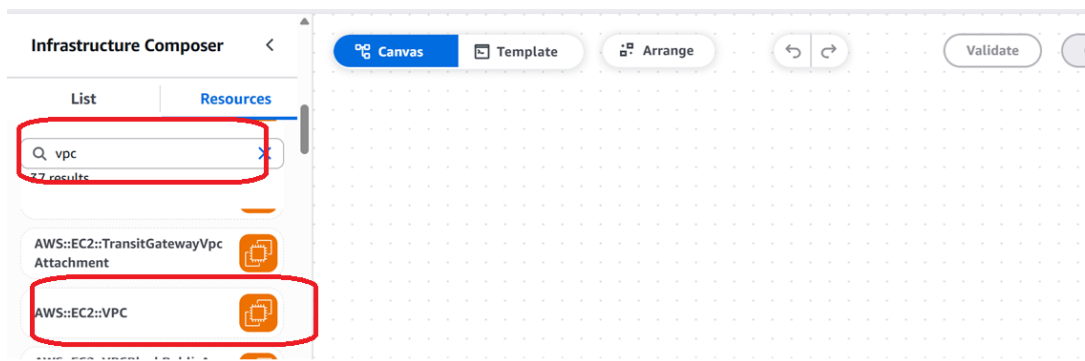
In aws console, select cloudformation:



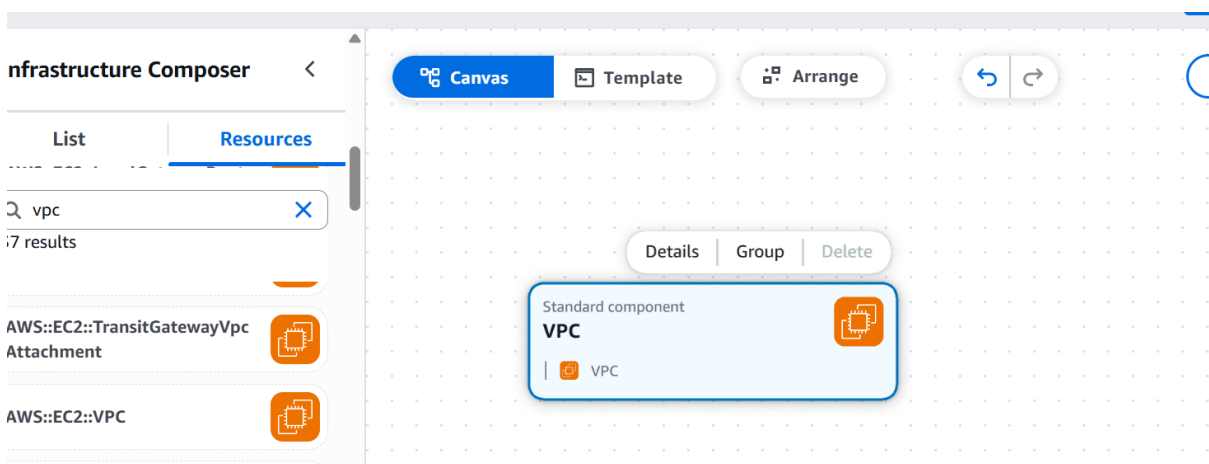
Select infrastructure composer



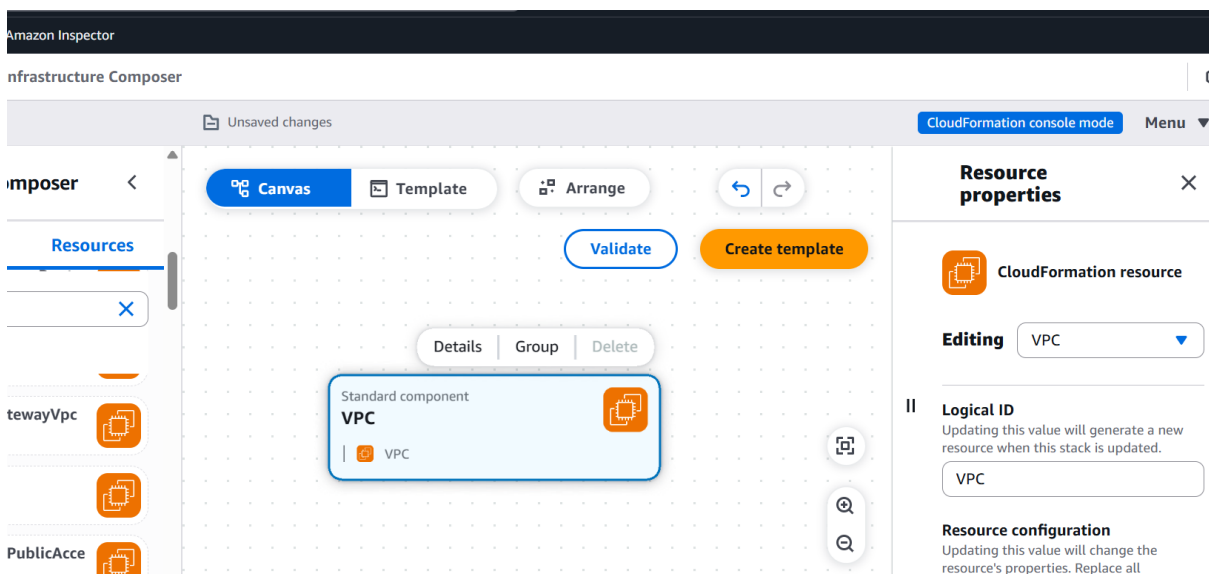
Search for VPC



Drag and drop vpc

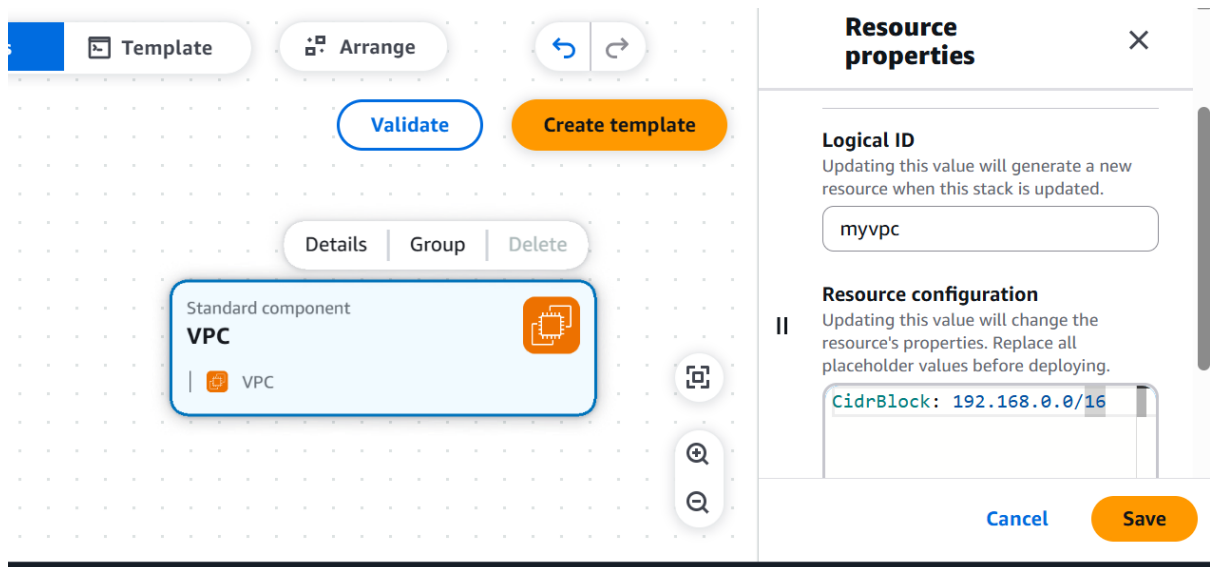


Double click on VPC



Right-hand side you will find Resource configuration

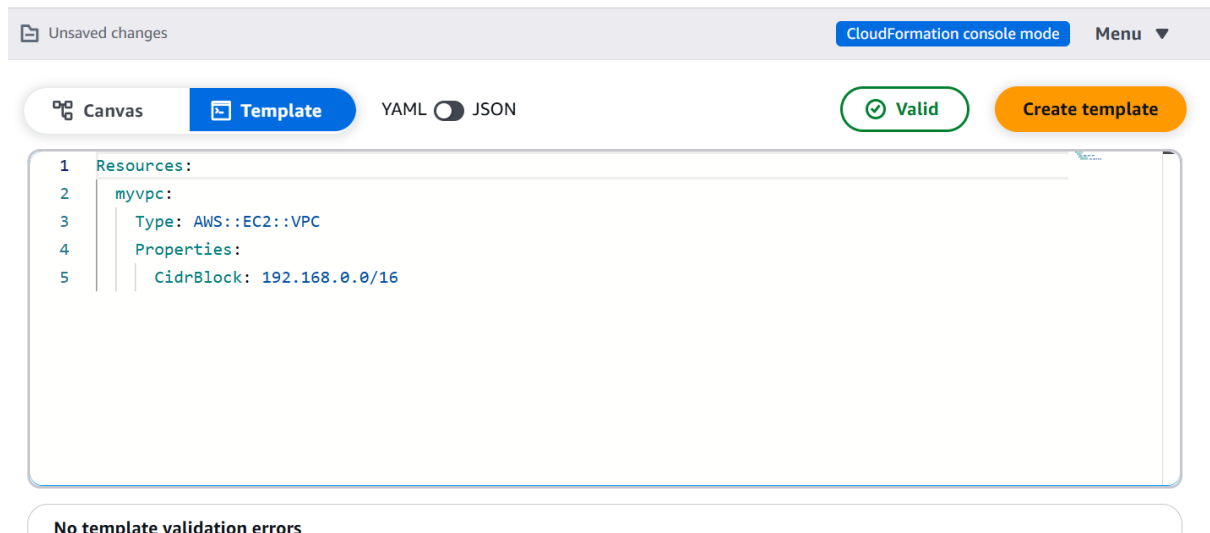
Add code in Resource configuration and save



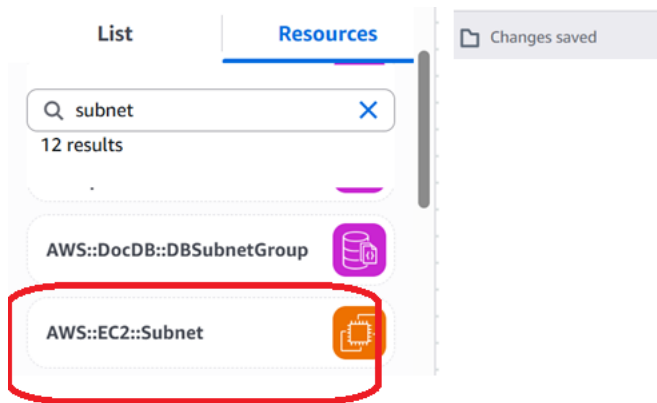
Add following:

CidrBlock: 192.168.0.0/16

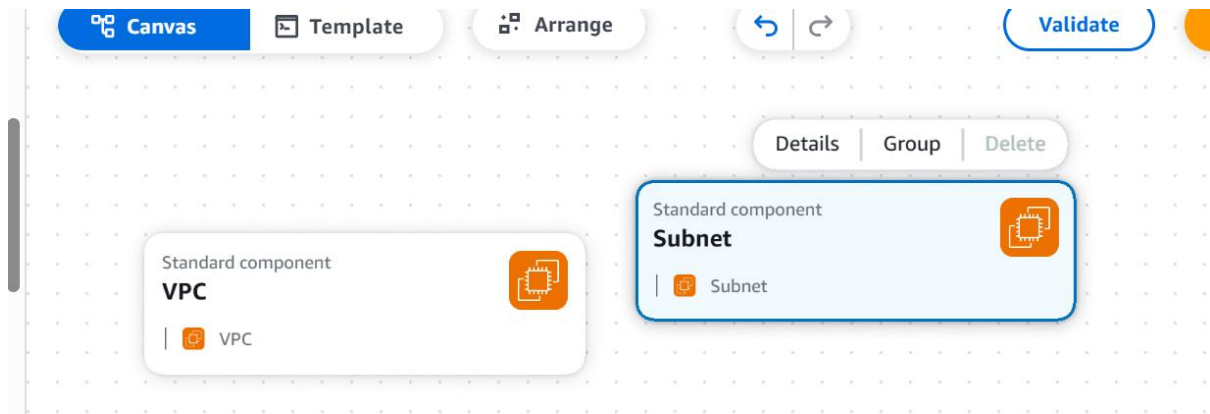
Click on Template you will find the code and click on validate and check



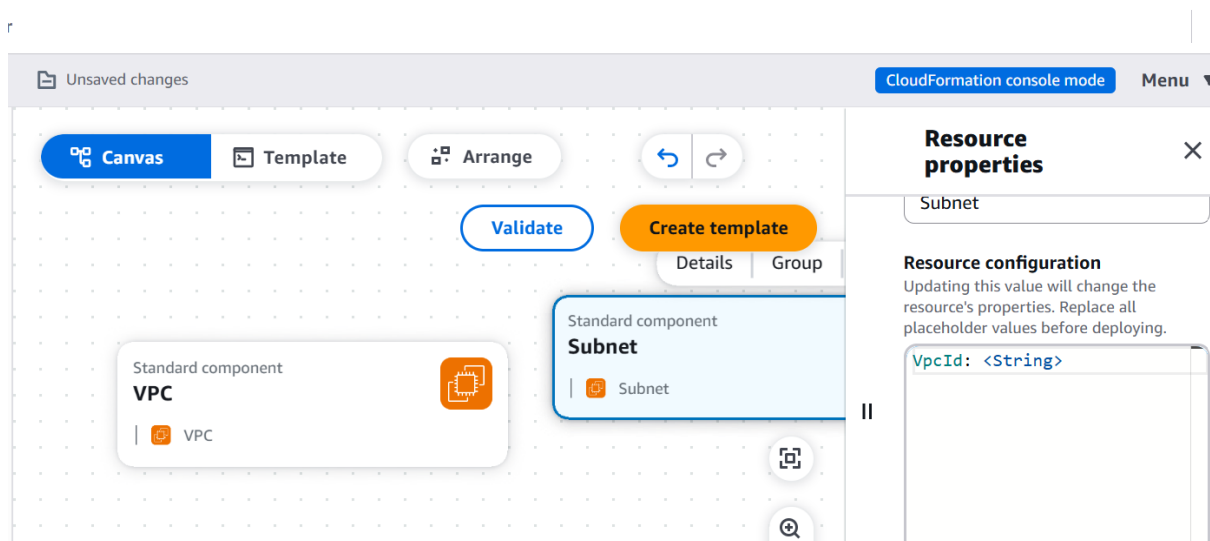
Check for subnet



Drag and drop on AWS::EC2::Subnet



Double click on Subnet and find resource Configuration



In **AWS CloudFormation**, `!Ref` (or `Ref`) is an **intrinsic function** used to **reference a resource, parameter, or pseudo parameter** within a CloudFormation template.

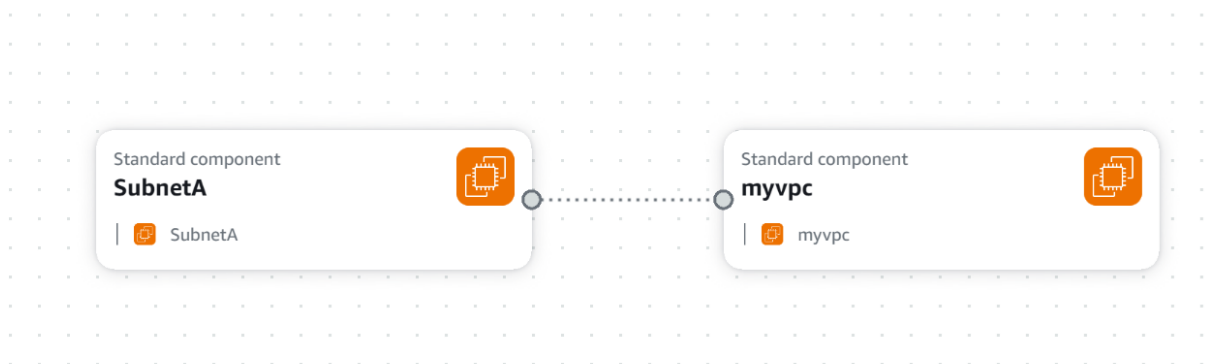
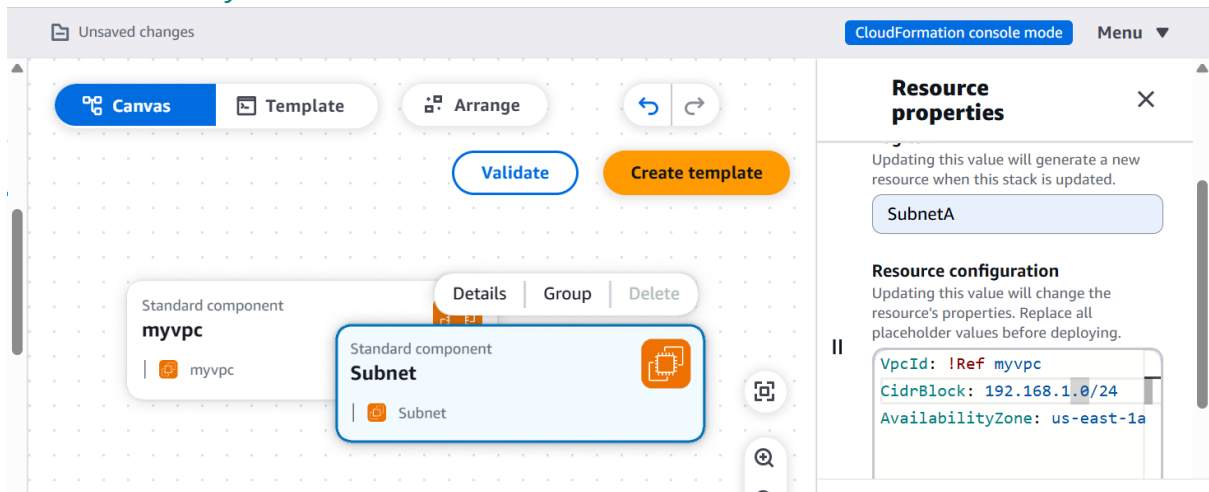
What does `!Ref` do?

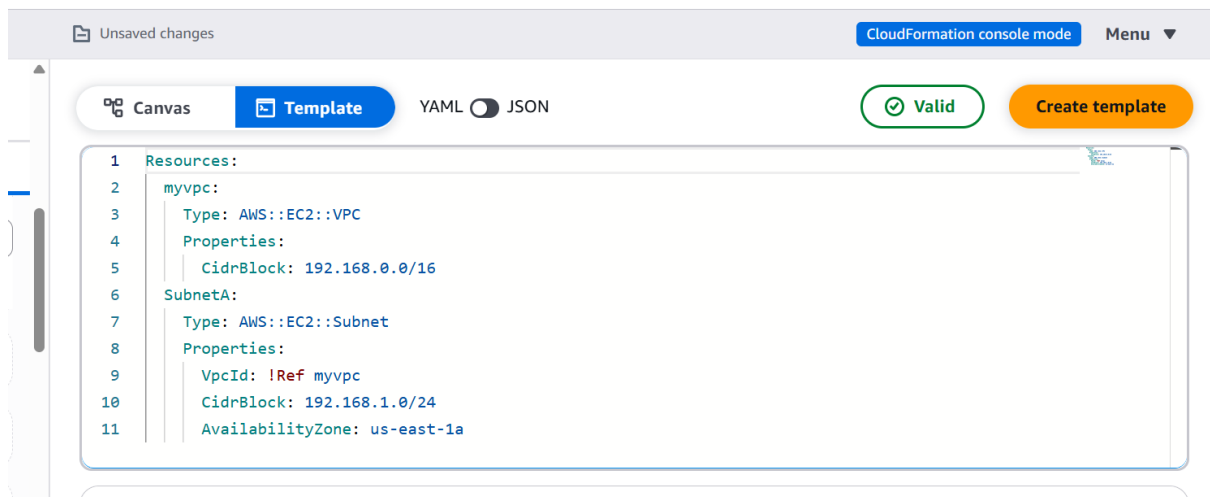
- It **returns a value** that's associated with the specified resource or parameter.
- The returned value depends on what is being referenced.

```
VpcId: !Ref myvpc
```

```
CidrBlock: 192.168.1.0/24
```

```
AvailabilityZone: us-east-1a
```



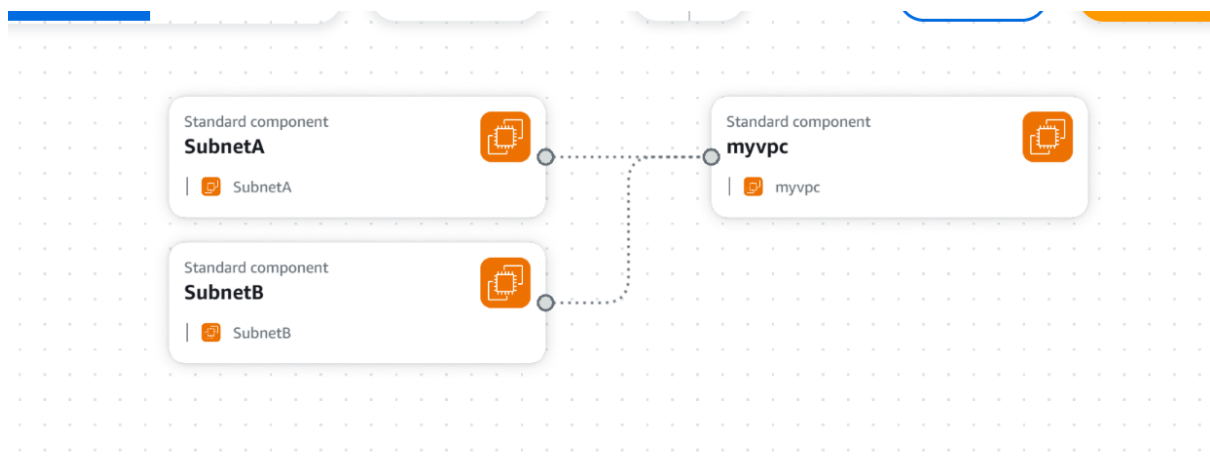


Assignment Create one more subnet

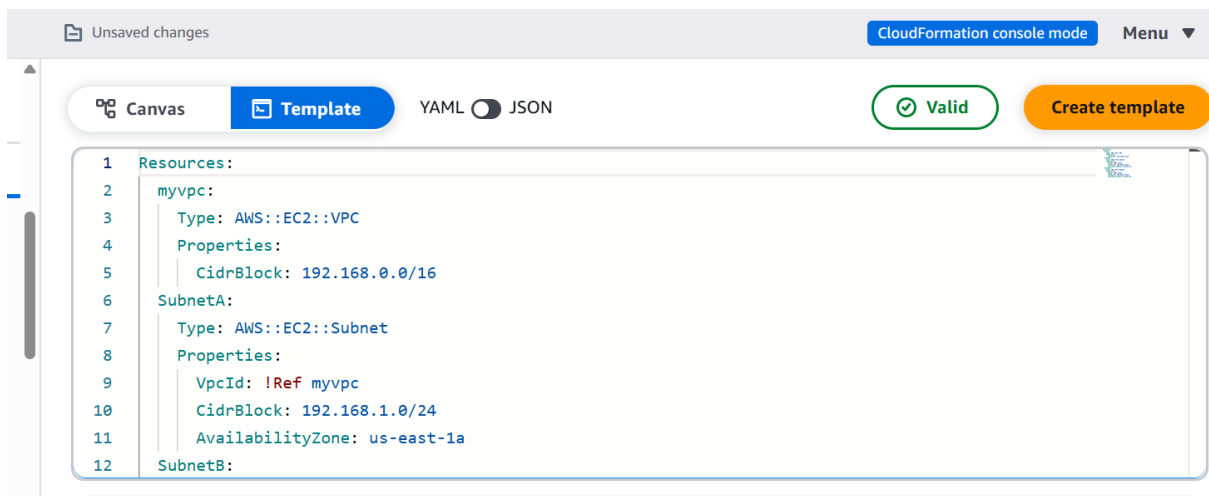
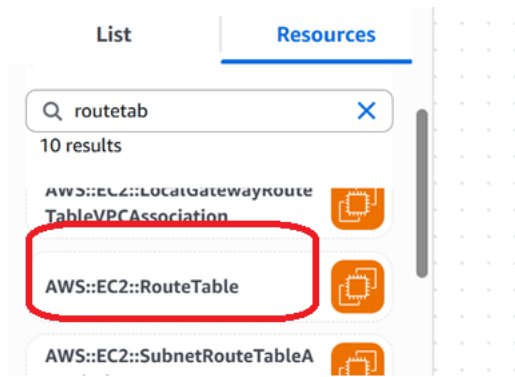
VpcId: !Ref myvpc

CidrBlock: 192.168.2.0/24

AvailabilityZone: us-east-1b

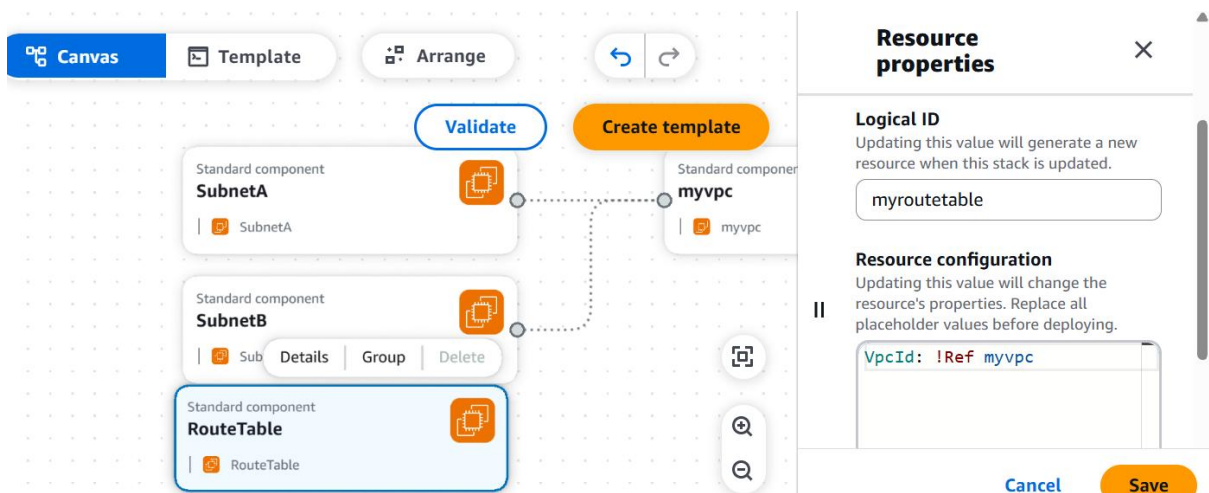


Create RouteTable



Drag and drop

VpcId: !Ref myvpc

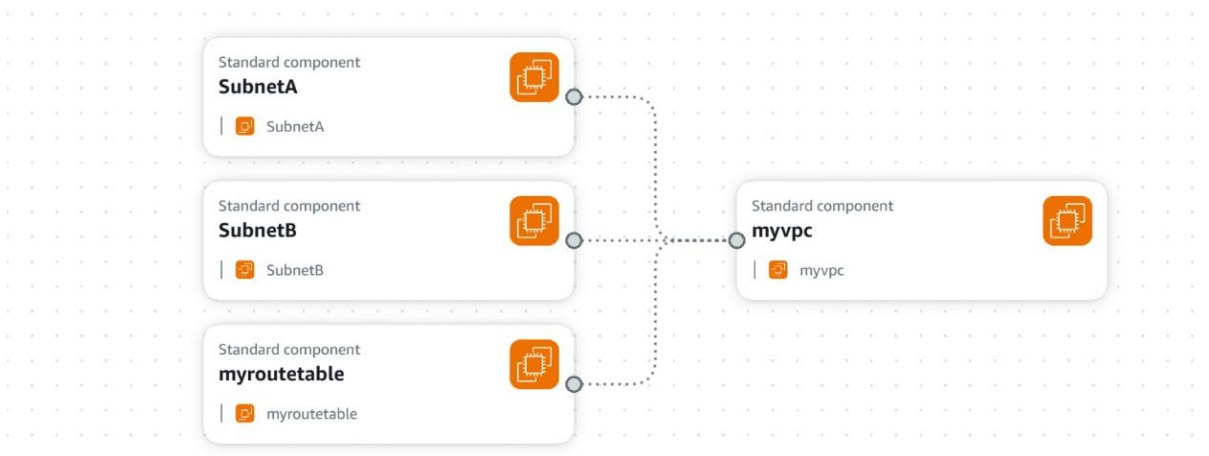


Save it

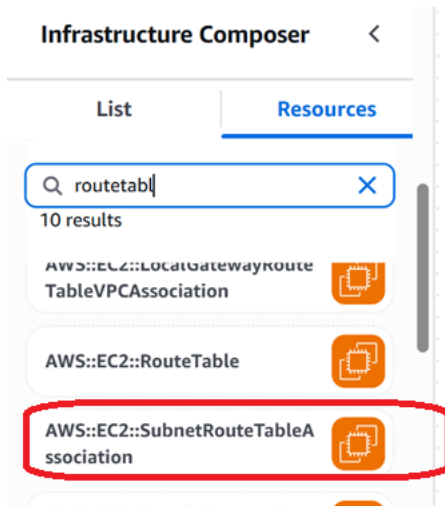
The screenshot displays the AWS CloudFormation console interface. At the top, a visual resource diagram shows three standard components on the left: **Subnet**, **Subnet2**, and **RouteTable**, and one on the right: **VPC**. Dotted lines indicate dependencies from each of the three left components to the **VPC** component. Below the diagram, the console header includes "Unsaved changes", "CloudFormation console mode", and a "Menu" dropdown. The main area has tabs for "Canvas" and "Template", with the "Template" tab selected. It also shows "YAML" and "JSON" format options, a "Valid" status button, and a "Create template" button. The YAML template content is as follows:

```
1 Resources:
2   myvpc:
3     Type: AWS::EC2::VPC
4     Properties:
5       CidrBlock: 192.168.0.0/16
6   SubnetA:
7     Type: AWS::EC2::Subnet
8     Properties:
9       VpcId: !Ref myvpc
10      CidrBlock: 192.168.1.0/24
11      AvailabilityZone: us-east-1a
12   SubnetB:
```

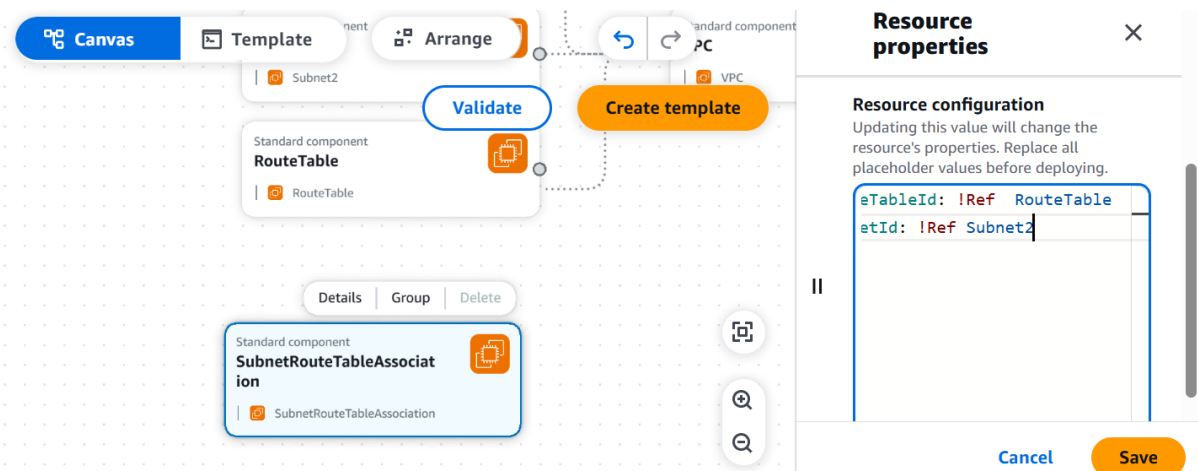
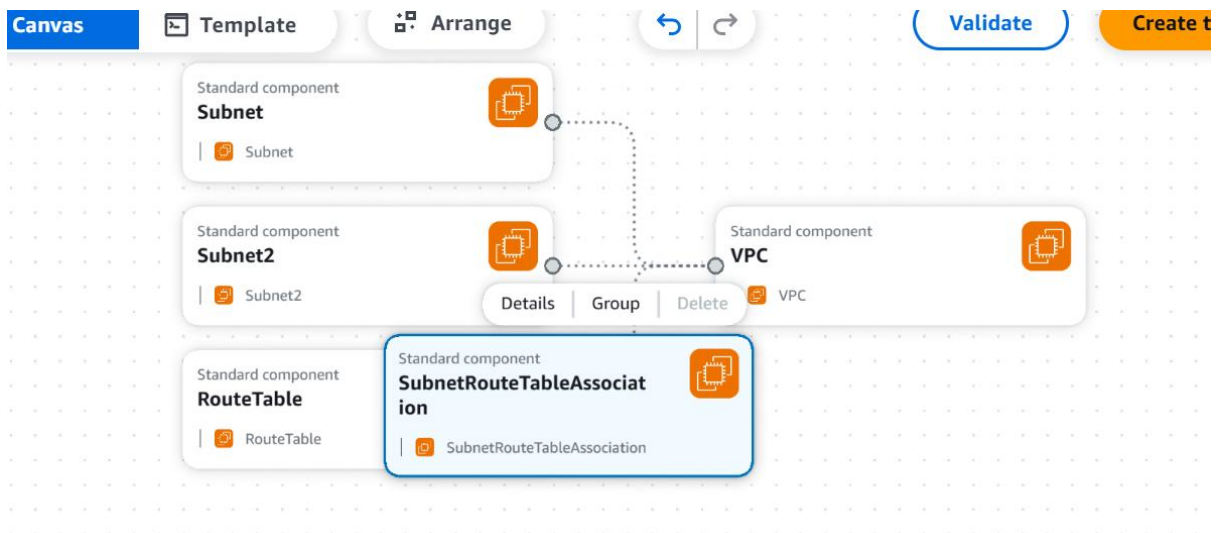
At the bottom of the console area, a message states: "No template validation errors".



Check for subnetroutetableassociation

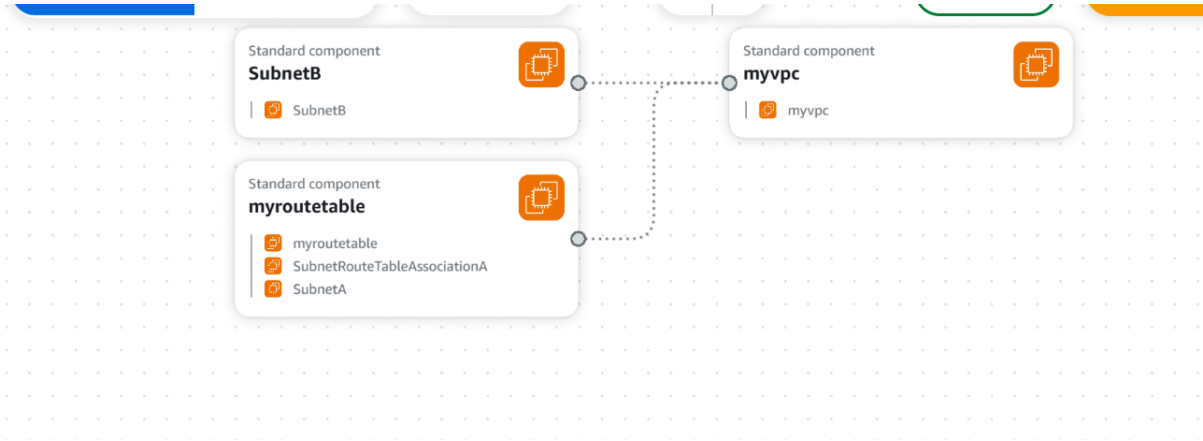
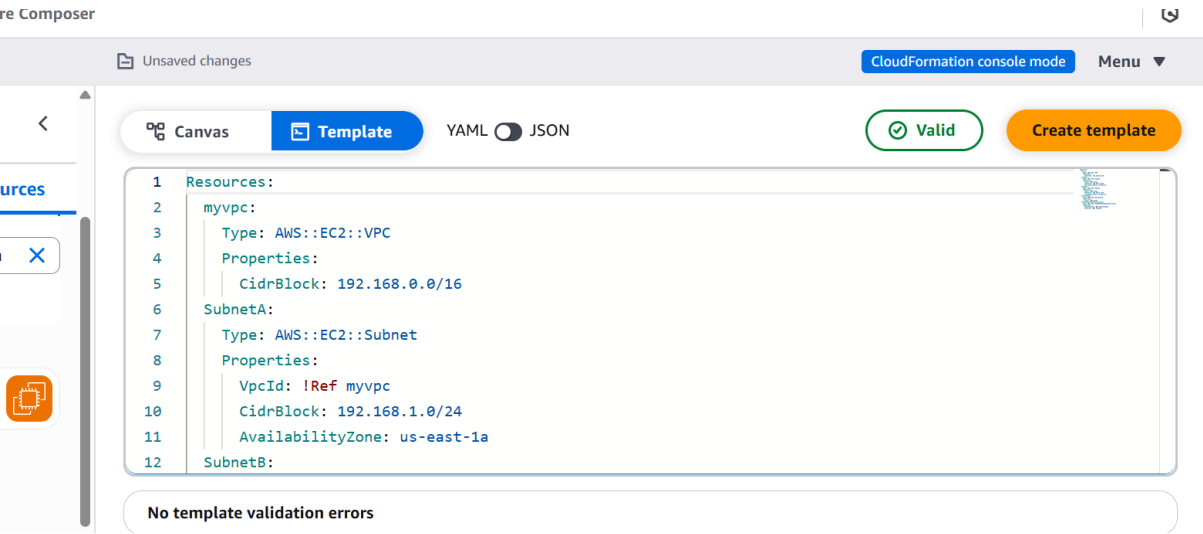
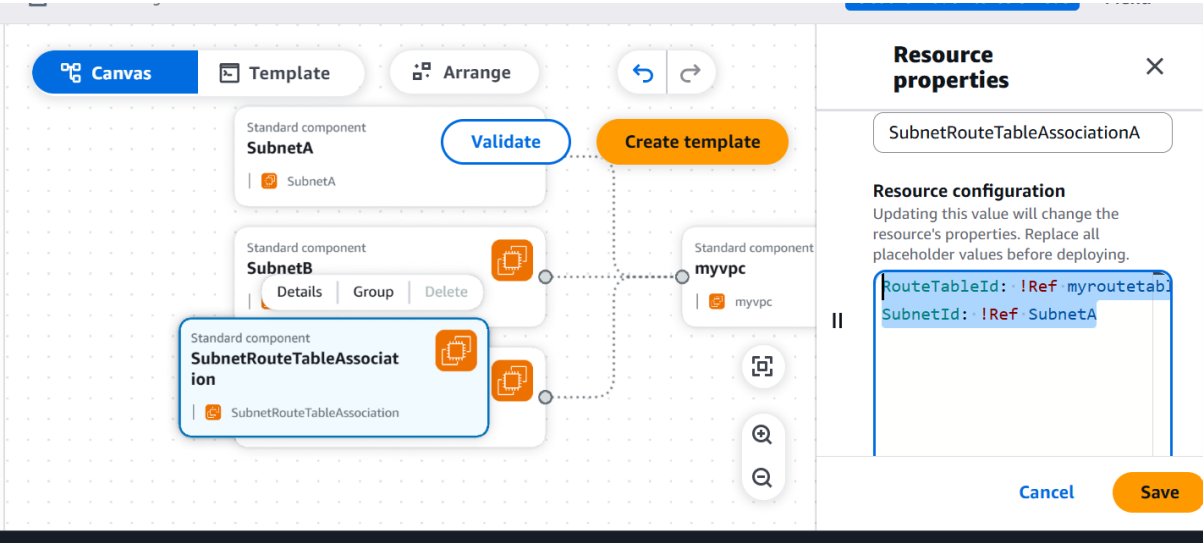


Drag and drop



RouteTableId: !Ref myroutetable

SubnetId: !Ref SubnetA



Assignment Add subnetB as well: using SubnetRouteTableAssociation

RouteTableId: !Ref myroutetable

SubnetId: !Ref SubnetB

Standard component **myroutetable**

- myroutetable
- SubnetRouteTableAssociationA
- SubnetA
- SubnetRouteTableAssociationB
- SubnetB

Standard component **myvpc**

- myvpc

Composer

Unsaved changes CloudFormation console mode Menu

Canvas Template YAML JSON Valid Create template

```
1 Resources:
2   myvpc:
3     Type: AWS::EC2::VPC
4     Properties:
5       CidrBlock: 192.168.0.0/16
6   SubnetA:
7     Type: AWS::EC2::Subnet
8     Properties:
9       VpcId: !Ref myvpc
10    CidrBlock: 192.168.1.0/24
11    AvailabilityZone: us-east-1a
12  SubnetB:
```

No template validation errors

Internetgateway:

Infrastructure Composer

List Resources

InternetGateway 2 results

▼ Standard IaC resources (2)

- AWS::EC2::EgressOnlyInternetGateway
- AWS::EC2::InternetGateway

Canvas Template Arrange

Standard component **myroutetable**

- myroutetable
- SubnetRouteTableAssociationA
- SubnetA
- SubnetRouteTableAssociationB
- SubnetB

Standard component **myvpc**

- myvpc

Drag and drop internetgateway

Unsaved changes

CloudFormation console mode

Canvas

Template

Arrange

Validate

Create t

es

X

Standard component

myroutetable

myroutetable

SubnetRouteTableAssociationA

SubnetA

SubnetRouteTableAssociationB

SubnetB

Standard component

myvpc

myvpc

Details

Group

Delete

Standard component

InternetGateway

InternetGateway

Canvas

Template

Arrange

Validate

Create template

Standard component

myroutetable

myroutetable

SubnetRouteTableAssociationA

SubnetA

SubnetRouteTableAssociationB

SubnetB

Standard component

myvpc

myvpc

Details

Group

Delete

Standard component

InternetGateway

InternetGateway

Resource properties

X

Logical ID

Updating this value will generate a new resource when this stack is updated.

myigw

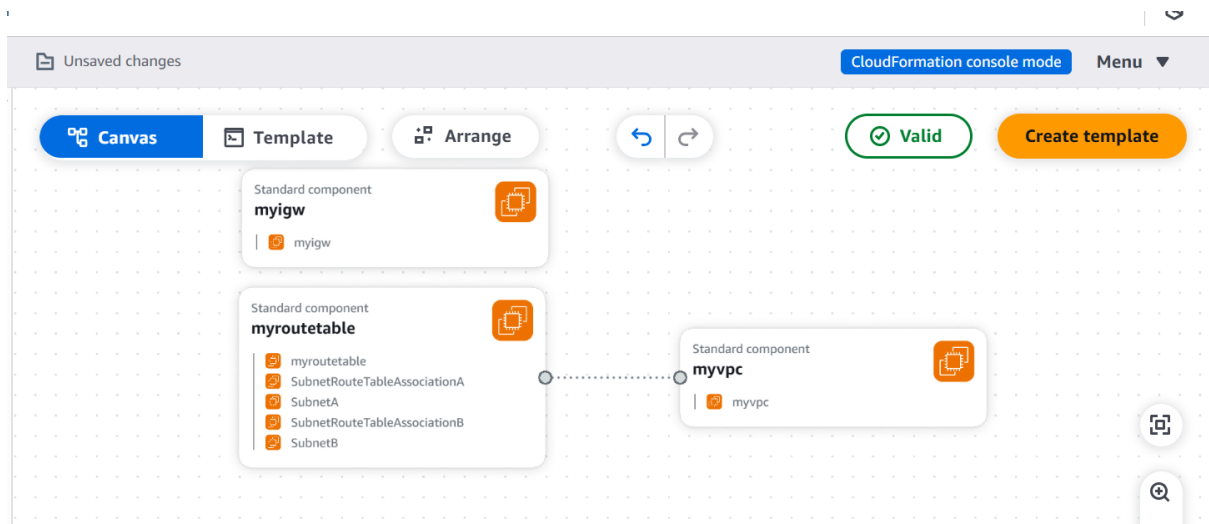
Resource configuration

Updating this value will change the resource's properties. Replace all placeholder values before deploying.

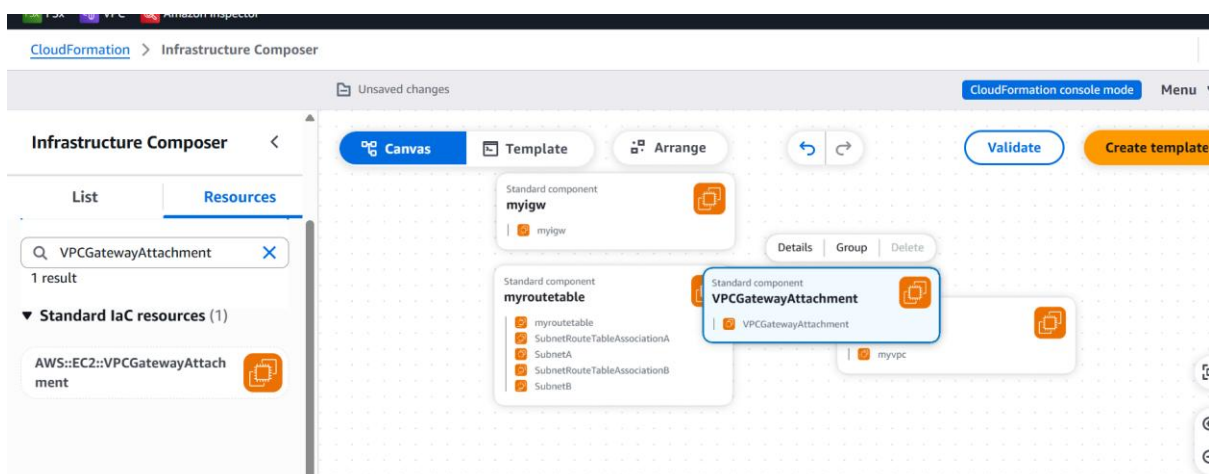
Cancel

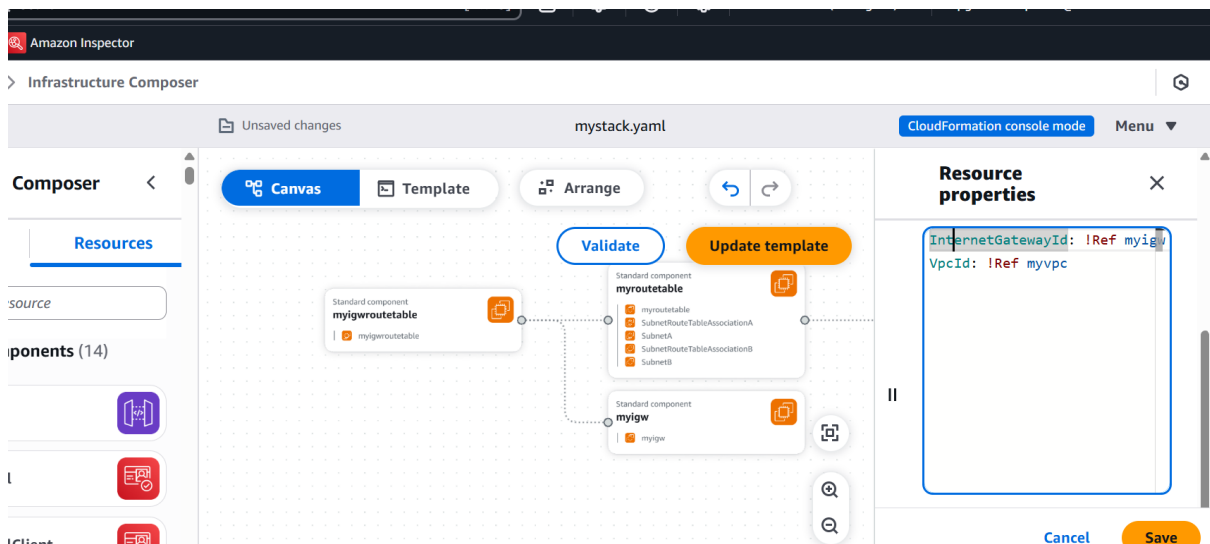
Save

Validate



Check for VPCGatewayAttachment:



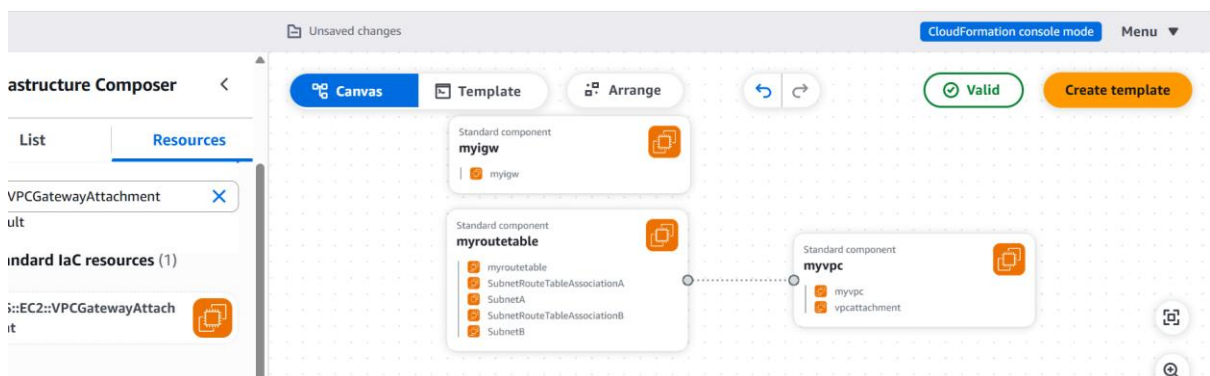


Code :

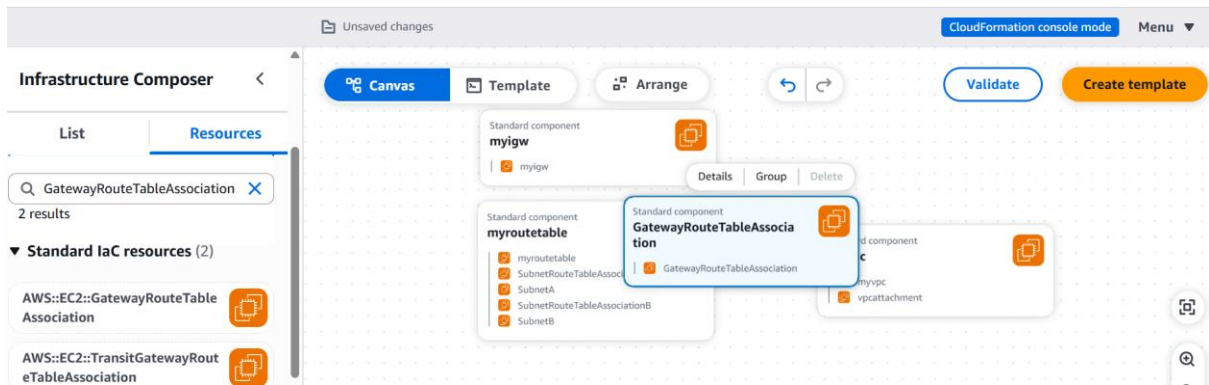
InternetGatewayId: !Ref myigw

VpcId: !Ref myvpc

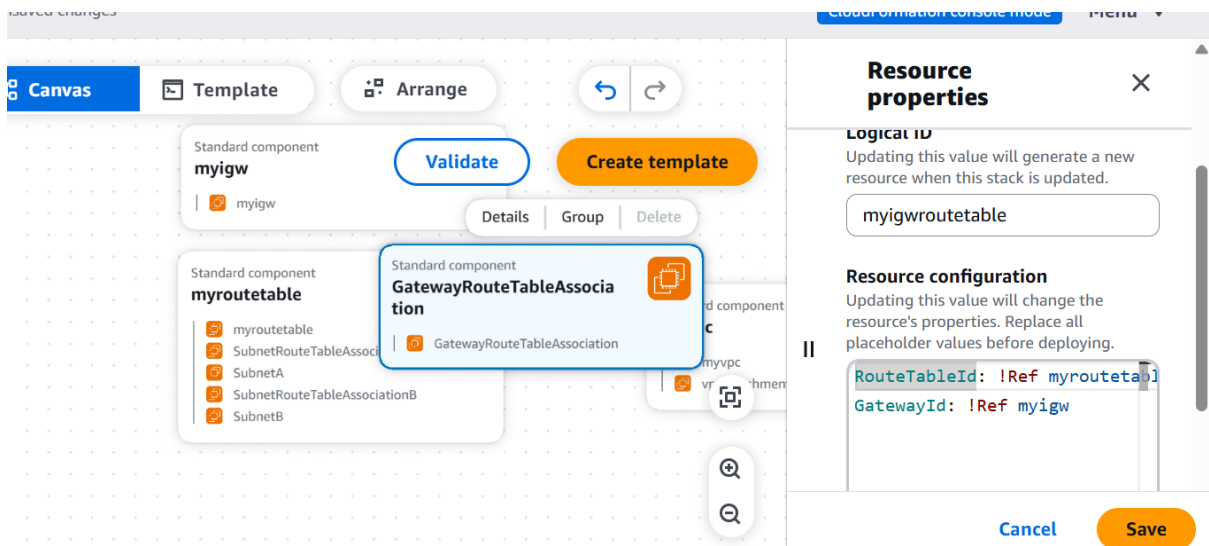
Save and validate



Next search for GatewayRouteTableAssociation



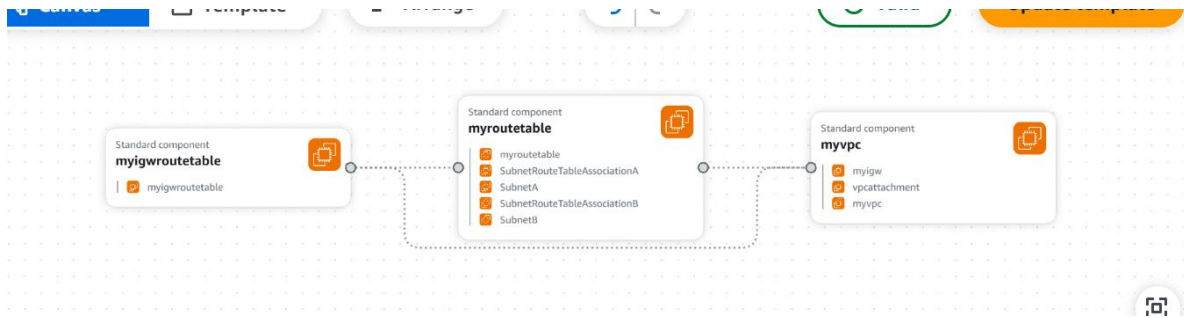
Double click and provide details



Code

RouteTableId: !Ref myroutetable

GatewayId: !Ref myigw



Click on validate

Code:

Resources:

myvpc:

Type: AWS::EC2::VPC

Properties:

CidrBlock: 192.168.0.0/16

SubnetA:

Type: AWS::EC2::Subnet

Properties:

VpcId: !Ref myvpc

CidrBlock: 192.168.1.0/24

AvailabilityZone: us-east-1a

SubnetB:

Type: AWS::EC2::Subnet

Properties:

VpcId: !Ref myvpc

CidrBlock: 192.168.2.0/24

AvailabilityZone: us-east-1b

myroutetable:

Type: AWS::EC2::RouteTable

Properties:

VpcId: !Ref myvpc

SubnetRouteTableAssociationA:

Type: AWS::EC2::SubnetRouteTableAssociation

Properties:

RouteTableId: !Ref myroutetable

SubnetId: !Ref SubnetA

SubnetRouteTableAssociationB:

Type: AWS::EC2::SubnetRouteTableAssociation

Properties:

RouteTableId: !Ref myroutetable

SubnetId: !Ref SubnetB

myigw:

Type: AWS::EC2::InternetGateway

vpcattachment:

Type: AWS::EC2::VPCGatewayAttachment

Properties:

InternetGatewayId: !Ref myigw

VpcId: !Ref myvpc

myigwroutetable:

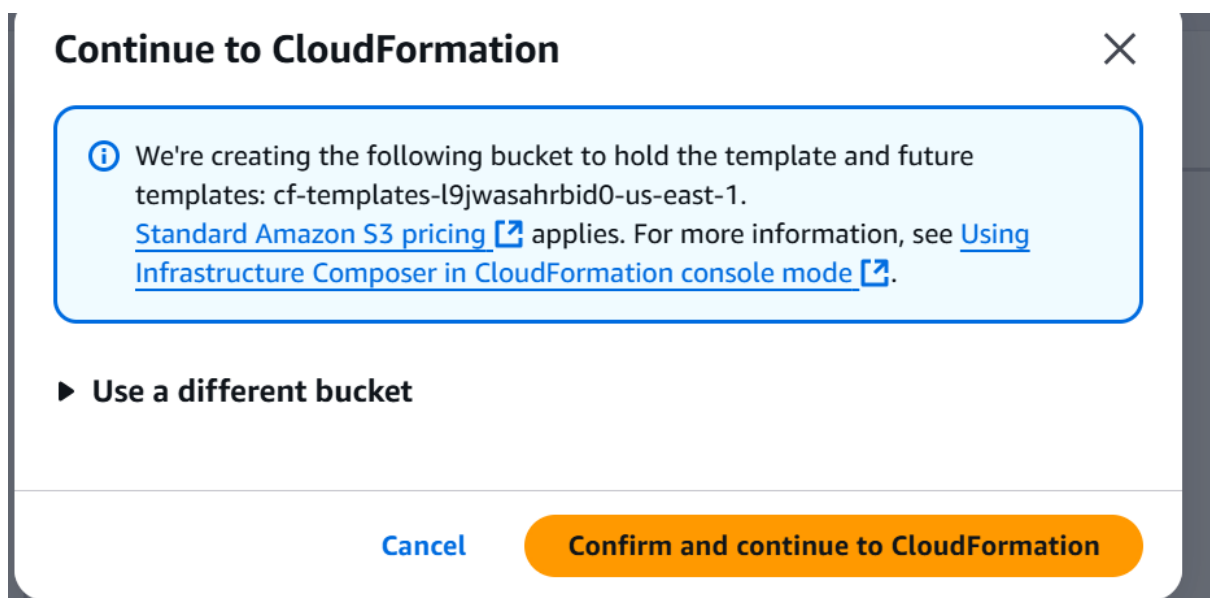
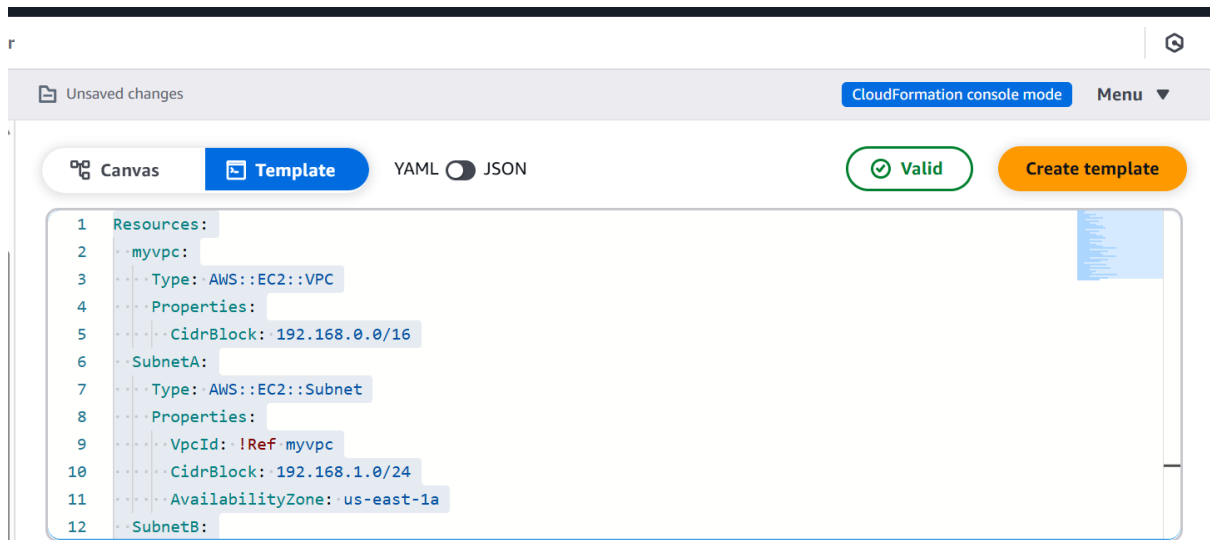
Type: AWS::EC2::GatewayRouteTableAssociation

Properties:

RouteTableId: !Ref myroutetable

GatewayId: !Ref myigw

[Click on create template](#)



Click on next

ack

view and create

☐ Choose an existing template
Upload or choose an existing template.

☒ Build from Infrastructure Composer
Create a template using a visual builder.

Create a template in Infrastructure Composer

Use Infrastructure Composer to visually design your stacks on a simple, drag-and-drop interface. Infrastructure Composer automatically updates and validates the template.

✔ Your template was successfully imported from Infrastructure Composer.

Amazon S3 URL
`https://s3.us-east-1.amazonaws.com/cf-templates-l9jwasahrbid0-us-east-1/template-17498716`

Edit in Infrastructure Composer

Cancel

Next

© 2025 Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

> Create stack

Step 2

☒ Specify stack details

Step 3

☐ Configure stack options

Step 4

☐ Review and create

Provide a stack name

Stack name
mystack

Stack name must contain only letters (a-z, A-Z), numbers (0-9) and hyphens (-) and start with a letter. Max 128 characters. Character count: 7/128.

Parameters

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

No parameters
There are no parameters defined in your template

Cancel

Previous

Next

No tags associated with the stack.

Add new tag

You can add 50 more tag(s)

Permissions - optional

Specify an existing AWS Identity and Access Management (IAM) service role that CloudFormation can assume.

IAM role – optional
Choose the IAM role for CloudFormation to use for all operations performed on the stack.

IAM role name ▼

Sample-role-name ▼

Remove

Stack failure options

Behaviour on provisioning failure

Specify the roll-back behaviour for a stack failure. [Learn more](#)

- ☒ **Roll back all stack resources**
Roll back the stack to the last known stable state.
- ☐ **Preserve successfully provisioned resources**
Preserves the state of successfully provisioned resources, while rolling back failed resources to the last known stable state. Resources without a last known stable state will be deleted upon the next stack operation.

Delete newly created resources during a rollback

Specify whether resources that were created during a failed operation should be deleted regardless of their deletion policy. [Learn more](#)

- ☐ **Use deletion policy**
Retains or deletes created resources according to their attached deletion policy.
- ☒ **Delete all newly created resources**
Deletes created resources during a rollback regardless of their attached deletion policy.

Click on submit

stack

Stack creation options

Timeout
-

Termination protection
Deactivated

Quick-create link

Use quick-create links to get stacks up and running quickly from the AWS CloudFormation console with the same basic configuration as this stack. Copy the URL on the link to share. [Learn more](#)

[Open quick-create link](#)

[Create changeset](#) [Cancel](#) [Previous](#) [Submit](#)

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

ⓘ Delete initiated for
arn:aws:cloudformation:us-east-
1:427998371624:stack/mystack
/7acd6050-48d1-11f0-a727-
0e7f81d39cf7

Stacks (2)

🔍 Filter by stack name

Filter status

Active



View nested

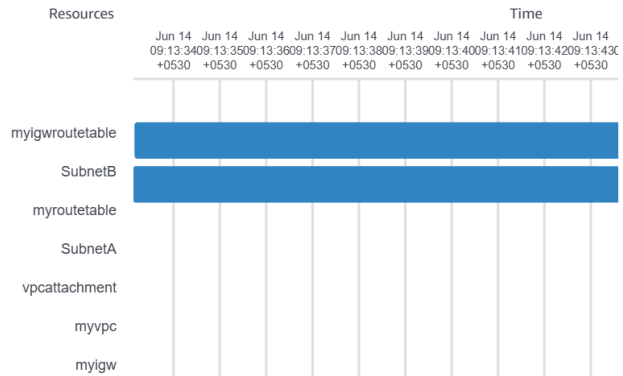
< 1 >

Stacks

myfirststack

deployment, this view will reset.

🔍 Search events



Search



ENG

09:13