

AWS CloudFormation Project

XYZ Technologies is a golden partner of AWS and utilizes most of the AWS services. Now, **CTO** is planning to **set up database infrastructure on the AWS cloud and expecting that infrastructure would expand in the future**. There can be multiple changes to infrastructure depending on the future requirements. Engineers should keep track of **future changes and easily roll back** to the previous version **if new infrastructure changes fail**.

Ample uses **MySQL** as a **database engine** and would like to create a database in the **private subnets**. You may use the below configurations to set up the complete infrastructure.

Database Configuration

Storage Space	20 GB
Database Instance Class	Db.t2.micro
Database name	Ampledbs
Database Engine	MySQL
Engine Version	8.0.23
PubliclyAccessible	False
Username	awsuser
Password	Aws123456789
Port	3306
AllowUpgradeVersion	False
BackUp Retention Period	7 Days

VPC Configuration

CIDR Block	192.168.0.0/16
EnableDNSHostnames	False
EnableDNSSupport	False
No. of Private Subnets	2
Private Subnet 1 CIDR	192.168.1.0/24
Private Subnet 1 AZ	us-east-1a
Private Subnet 2 CIDR	192.168.2.0/24
Private Subnet 2 AZ	us-east-1b

Security Group Configuration

SG Ingress CIDR	0.0.0.0/0
SG Ingress FromPort	3306
SG Ingress ToPort	3306
SG Ingress IPProtocol	TCP
SG Egress CIDR	0.0.0.0/0
SG Egress FromPort	80
SG Egress ToPort	80
SG Egress IPProtocol	TCP

To-Dos:

- List down AWS services to be used? – CloudFormation, VPC, Database - MySQL
- Suggest your approach.
- Design your plan
- Implement your plan
- Verify if the complete infrastructure is running

Deliverables:

- Setup Complete Infrastructure

KEY POINTS

CloudFormation: CloudFormation stack to deploy an EC2 instance. CloudFormation is a powerful service that allows you to manage your AWS Infrastructure as a Code (IaC), making it easier to provision and manage your resources.

CloudFormation is a service that allows you to create and manage AWS resources using code rather than provisioning them manually on AWS Management Console. This offers several benefits such as consistent and repeatable deployments Version Control and change Management, automatic roll backs and error handling, and the ability to create reusable templates for different environments.

By using CloudFormation, you can define your entire AWS Infrastructure including resources like EC2 instances, security groups, and more in a template file. This template can then be used to provision and manage those resources, ensuring that your infrastructure is deployed consistently every time.

CTO: Chief Technical Officer

Database: A structured collection of data, organized for efficient retrieval and management. It's essentially a way to store and access information, whether it's electronic or physical

VPC: It stands for Virtual Private Cloud. A VPC is a virtual network that closely resembles a traditional network that you would operate in your own data center. After you create a VPC, you can add subnets.

Subnet: A subnet is a range of IP addresses in your VPC. A subnet must reside in a single Availability Zone. After you add subnets, you can deploy AWS resources in your VPC.

Private Subnet: A private subnet is a subnet within a Virtual Private Cloud (VPC) that does not have a direct route to the internet. Resources within a private subnet can't be directly accessed from the public internet, enhancing security by limiting exposure. They typically rely on Network Address Translation (NAT) gateways or instances in public subnets to access the internet for outbound traffic.

Public Subnet: A public subnet in a Virtual Private Cloud (VPC) is a subnet that has a route to an Internet Gateway, allowing resources within the subnet to directly communicate with the public internet. This direct access is enabled by a route table that contains a route to the Internet

Gateway. Essentially, it is a subnet where resources can be accessed from and send traffic to the internet.

CIDR: It stands for Classless Inter-Domain Routing. It is an IP address allocation method that improves data routing efficiency on the internet. Every machine, server, and end-user device that connects to the internet has a unique number, called an IP address, associated with it. Devices find and communicate with one another by using these IP addresses. Organizations use CIDR to allocate IP addresses flexibly and efficiently in their networks

Ingress rule: A configuration that controls incoming network traffic to a system or application.

Egress rule: An egress rule defines the permitted outbound traffic from a network or system

PART 1: AWS SERVICES USED

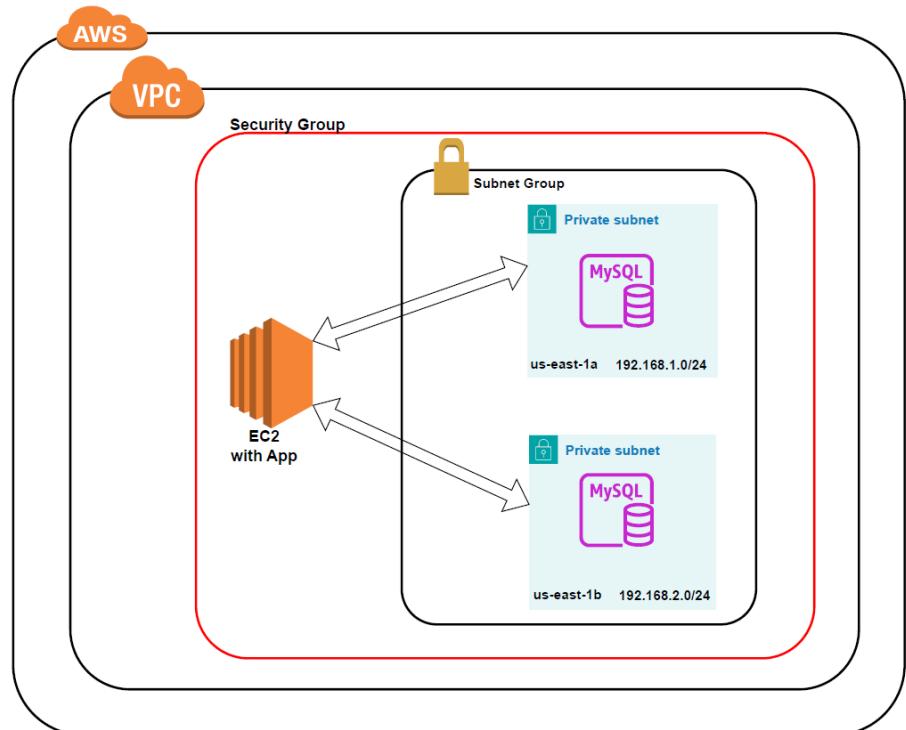
In this project, I will do the following:

1. Designed and implemented infrastructure-as-code using AWS CloudFormation.
2. Created templates for rapid deployment of complex applications.
3. Enforced best practices for reliable and scalable infrastructure provisioning.

We will use the following AWS Services

- CloudFormation
- VPC: This is the father of security groups
- Amazon RDS: RDS is running on EC2, since a database always need a server.
- Security Group that is a child to VPC
- Two Private Subnets
- Subnet Group (Used when dealing with RDS): It is advisable to create one Subnet Group and add the two subnets in it.

Architecture

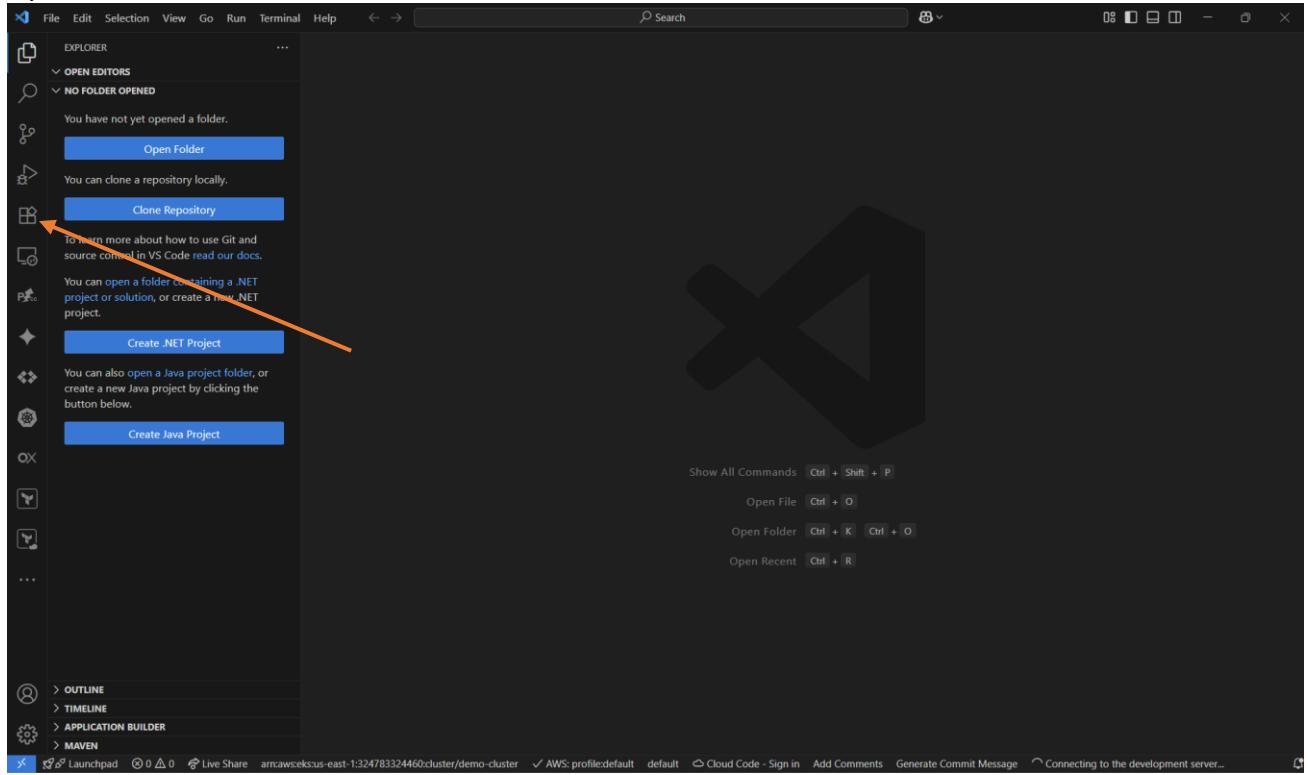


Follow these steps to create a DB instance in a VPC:

- Step 1: Create a VPC
- Step 2: Create a DB subnet group for the two private subnets
- Step 3: Create a VPC security group
- Step 4: Create a DB instance in the VPC

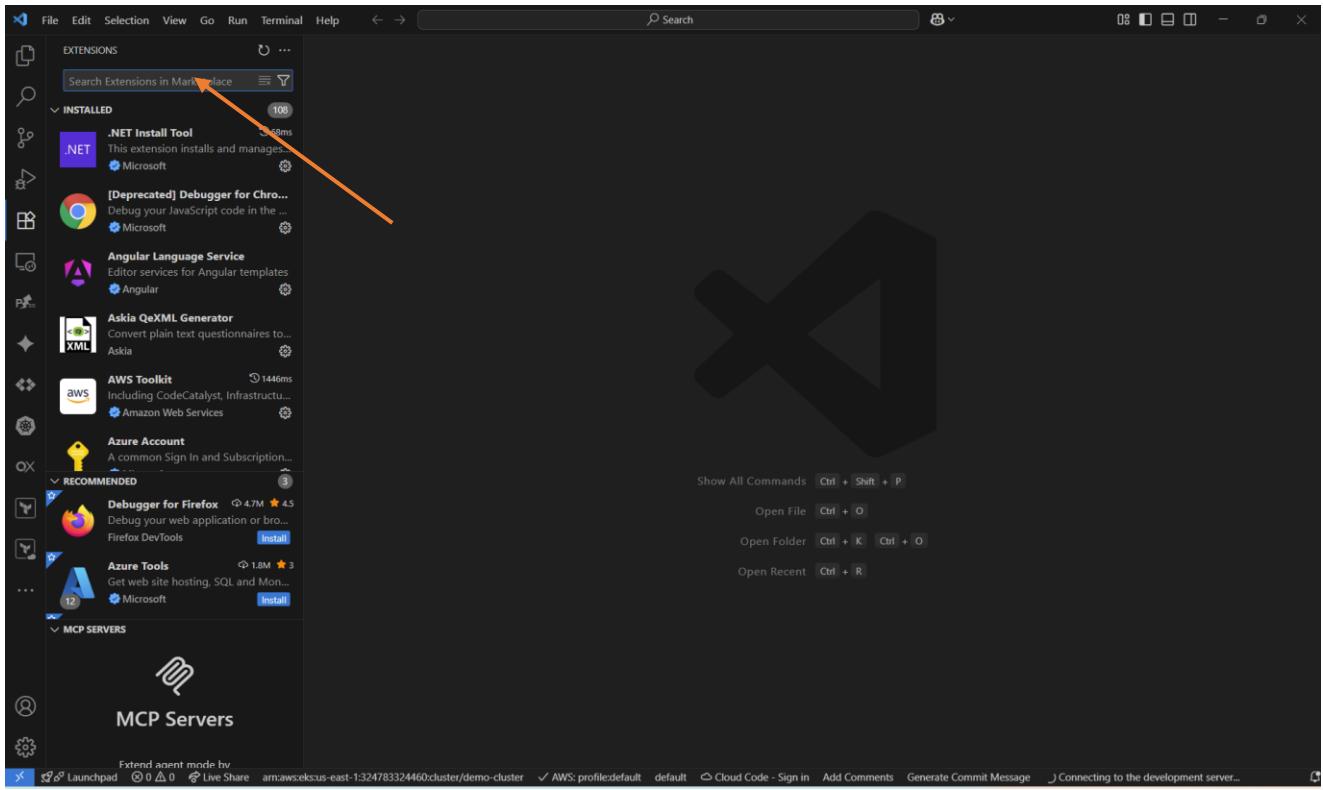
PART 4: IMPLEMENTATION

Open VSCode and install the extensions for CloudFormation

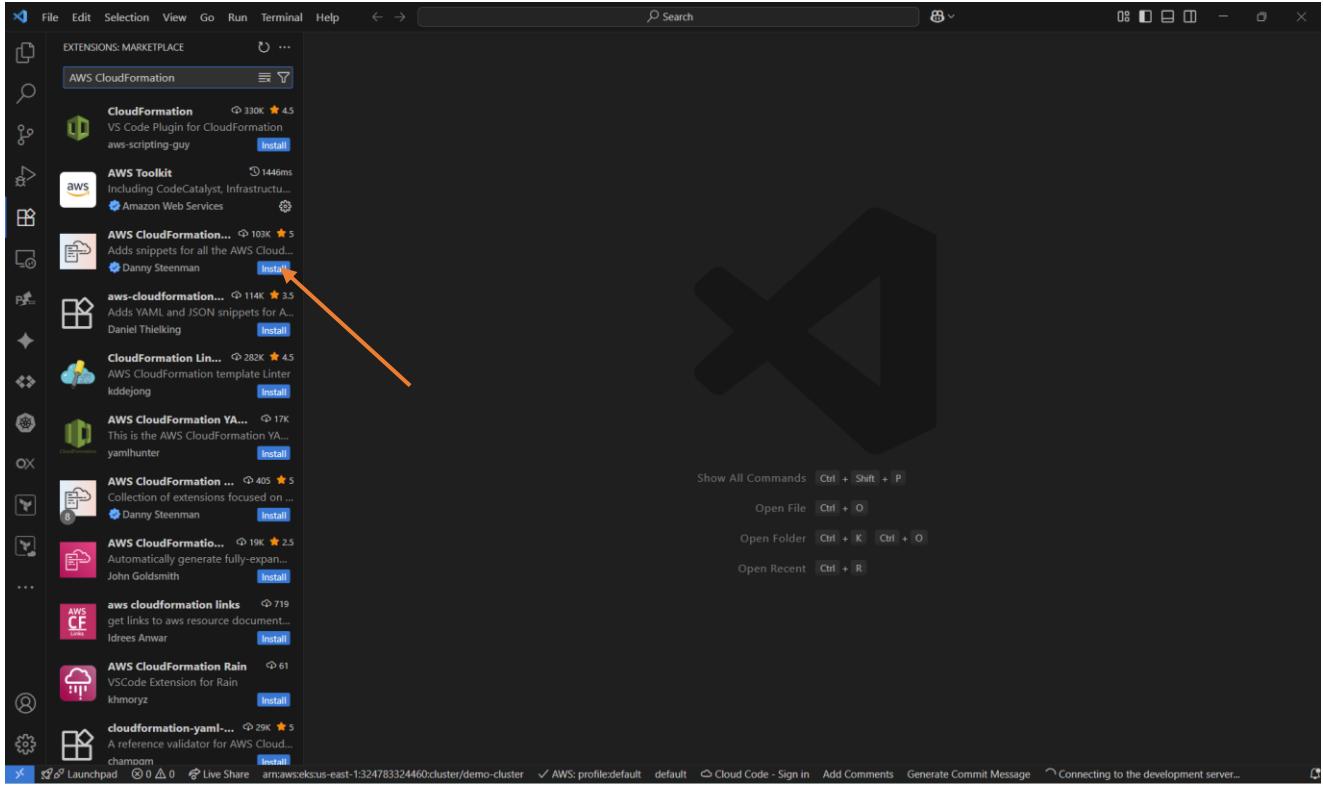


Click on “Extensions”

Prepared by Sidney Smith Ebot

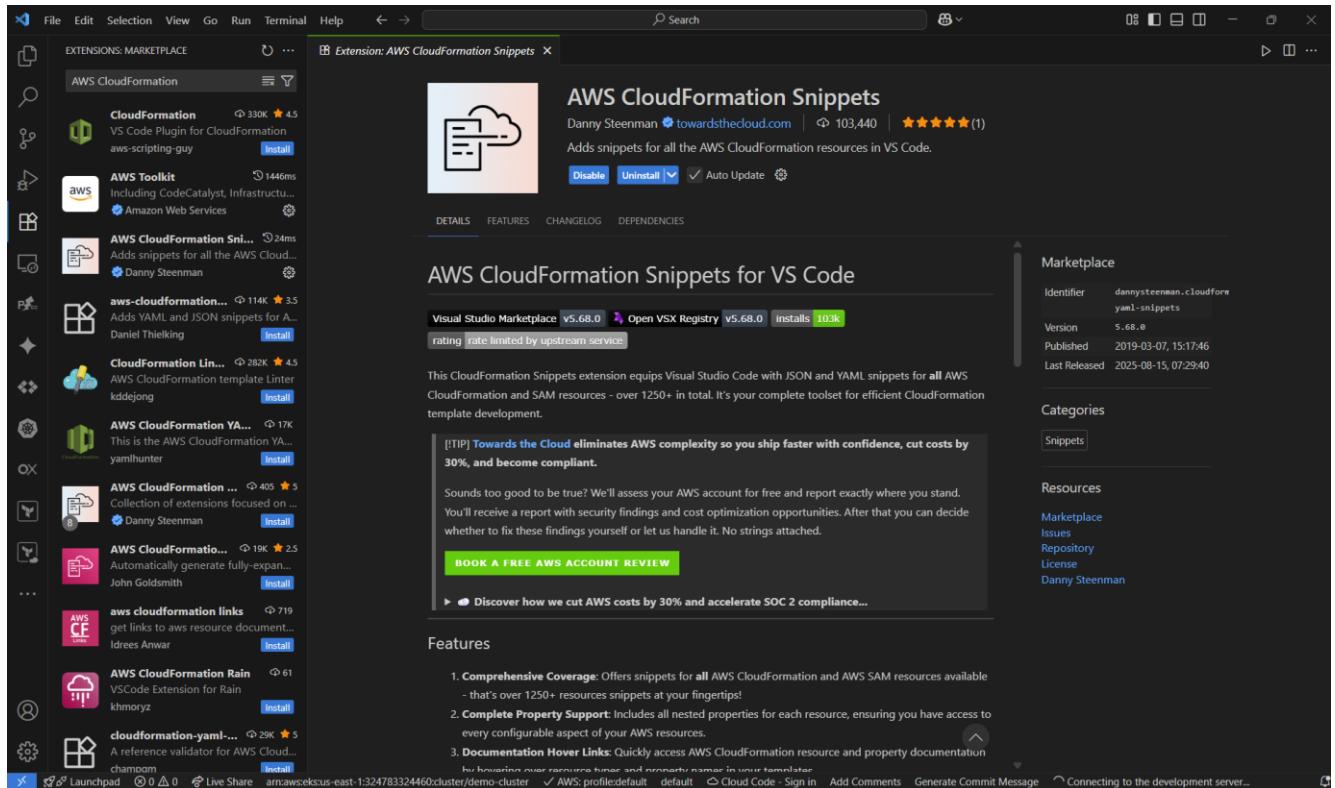


Search for “AWS CloudFormation”



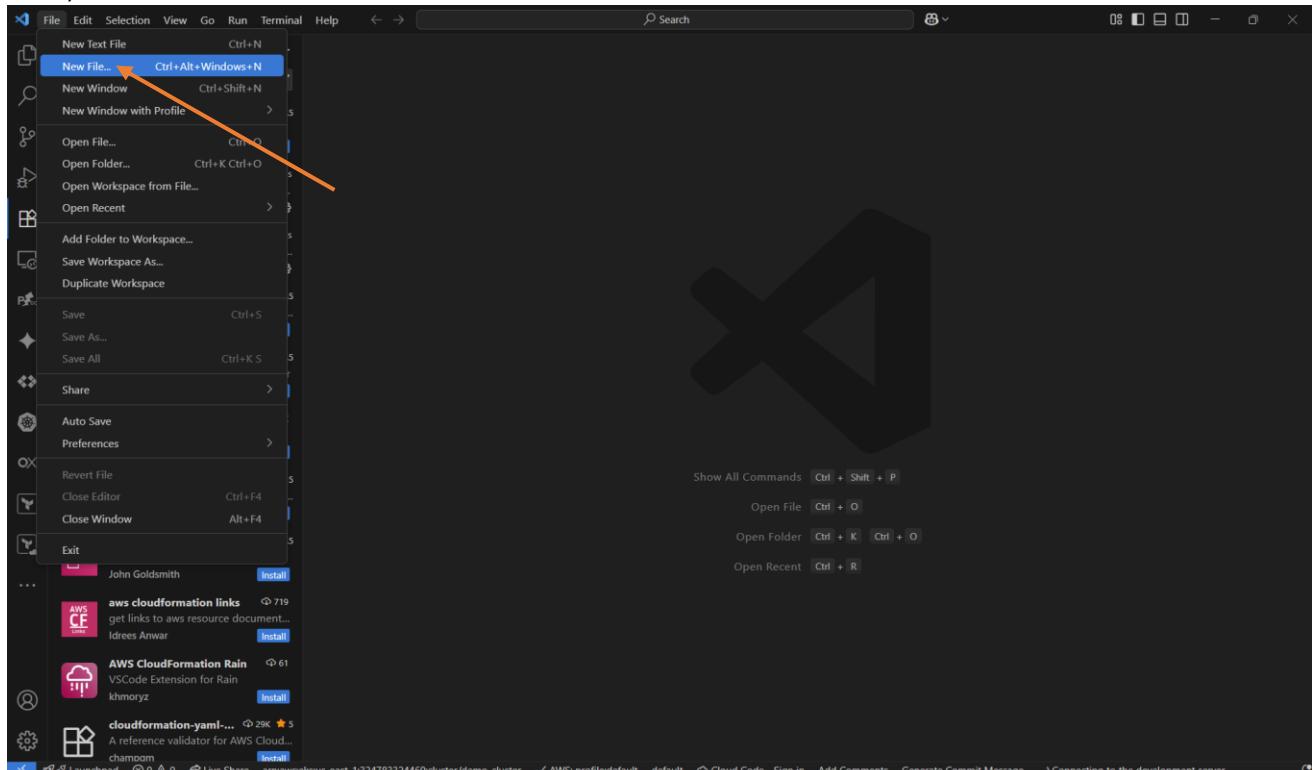
Click on install on “AWS CloudFormation Snippet”

Prepared by Sidney Smith Ebot



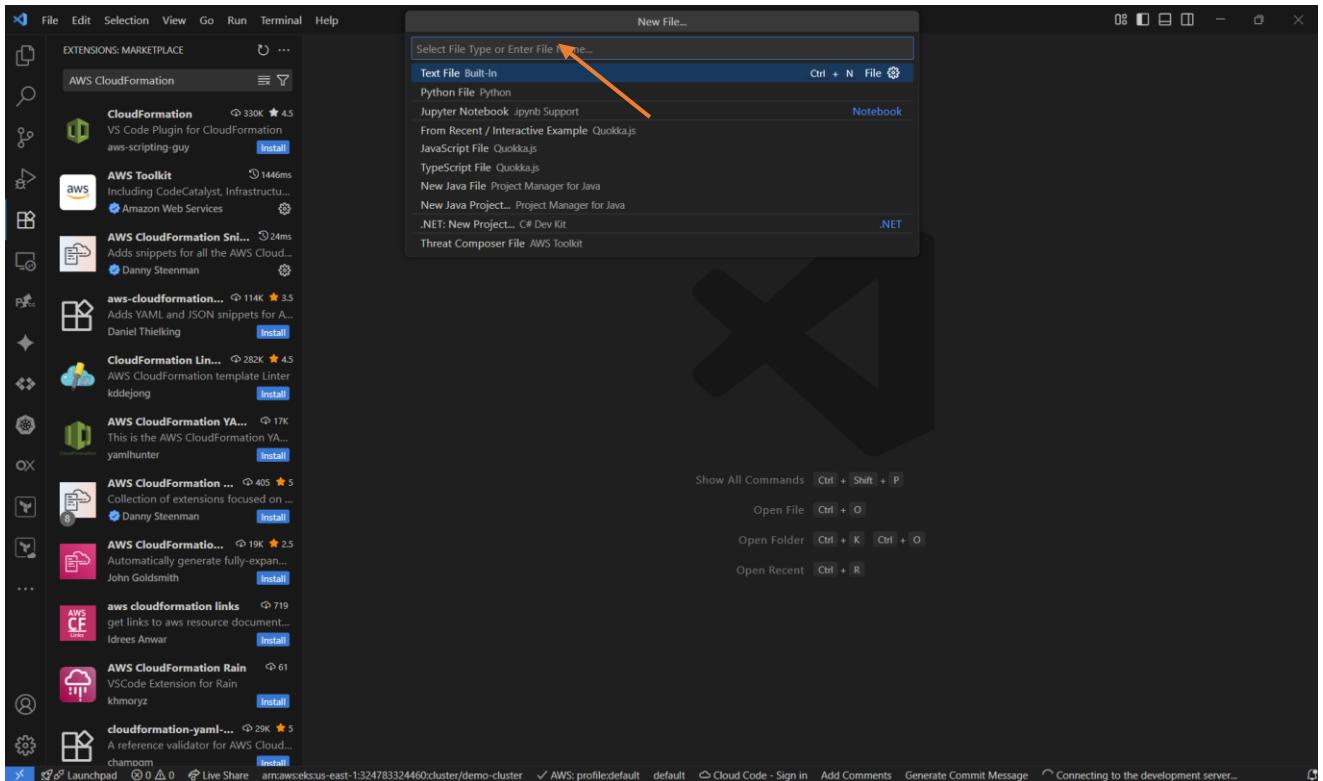
You can see that the “AWS CloudFormation Snippet” has been installed.

Now, click on “File”

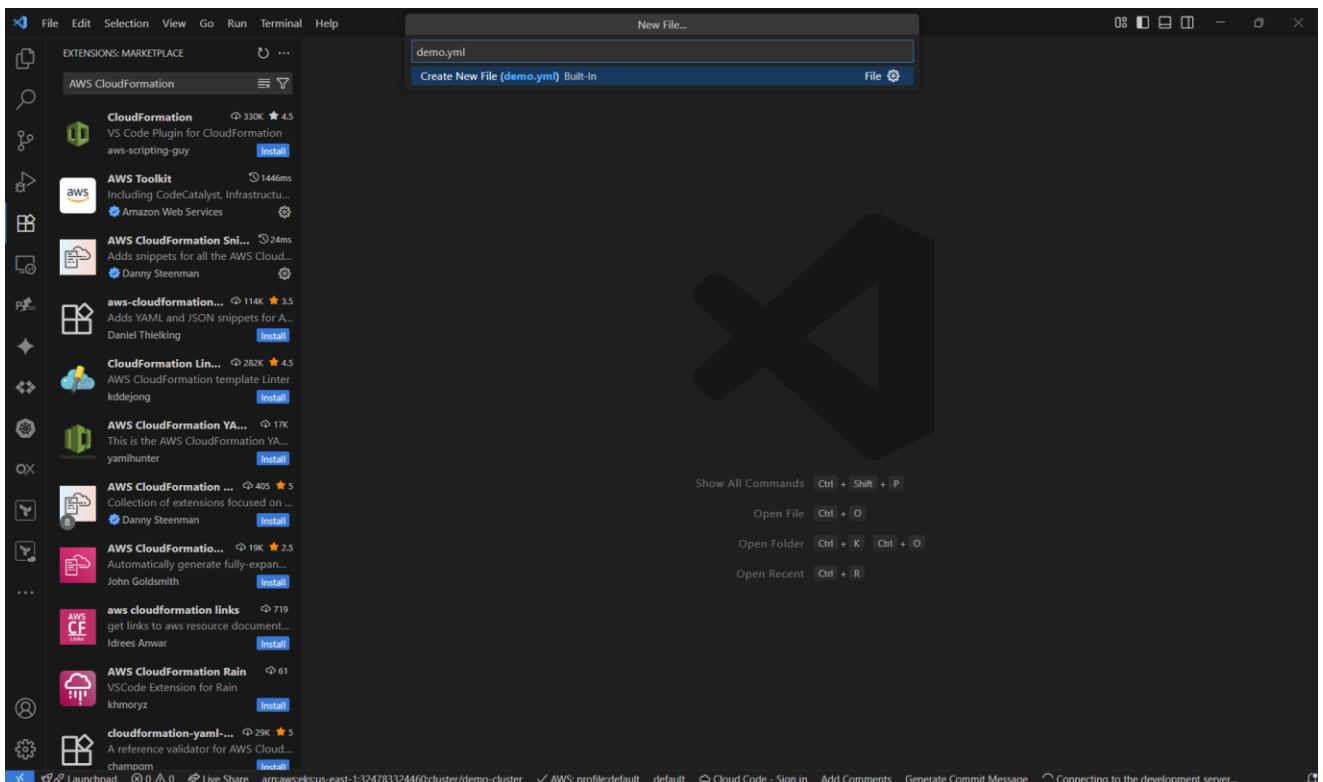


Select “New File”

Prepared by Sidney Smith Ebot

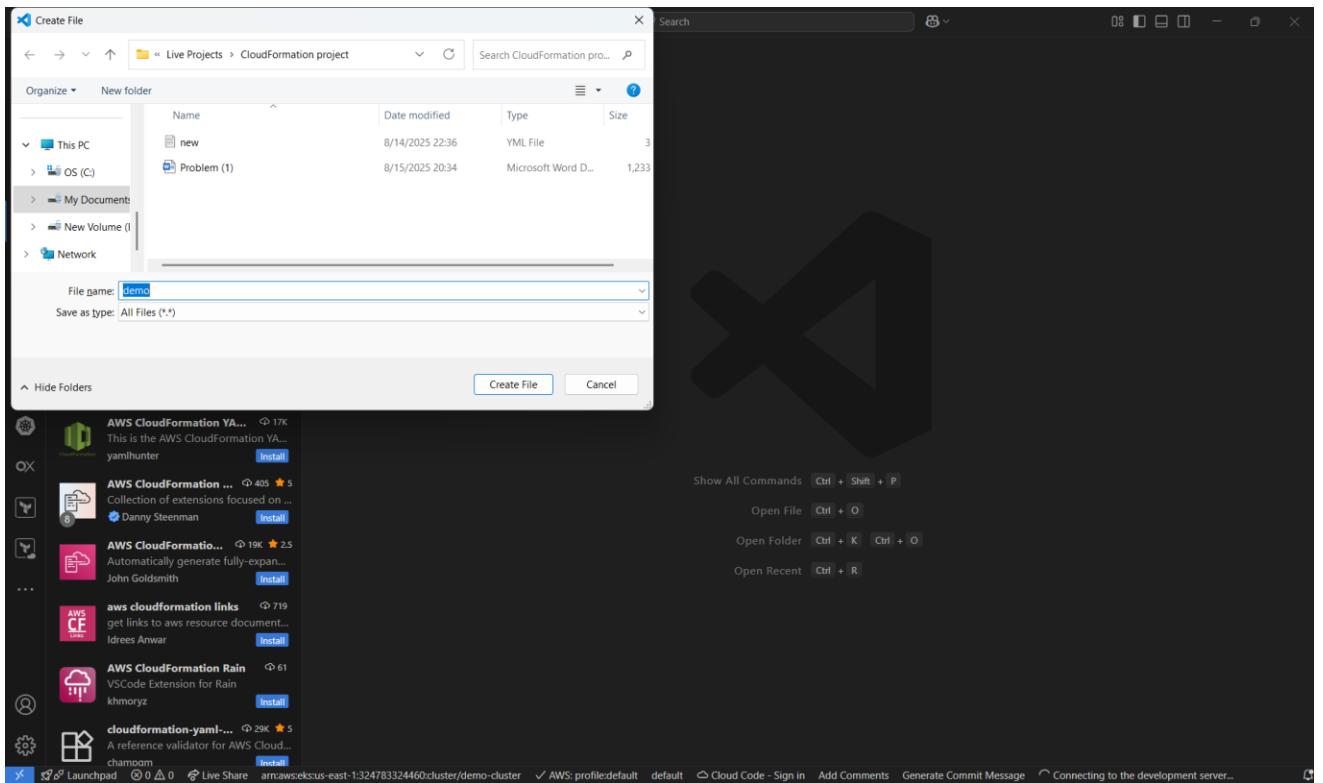


Enter “**demo.yml**”. “**demo**” is the name of our CloudFormation configuration file.

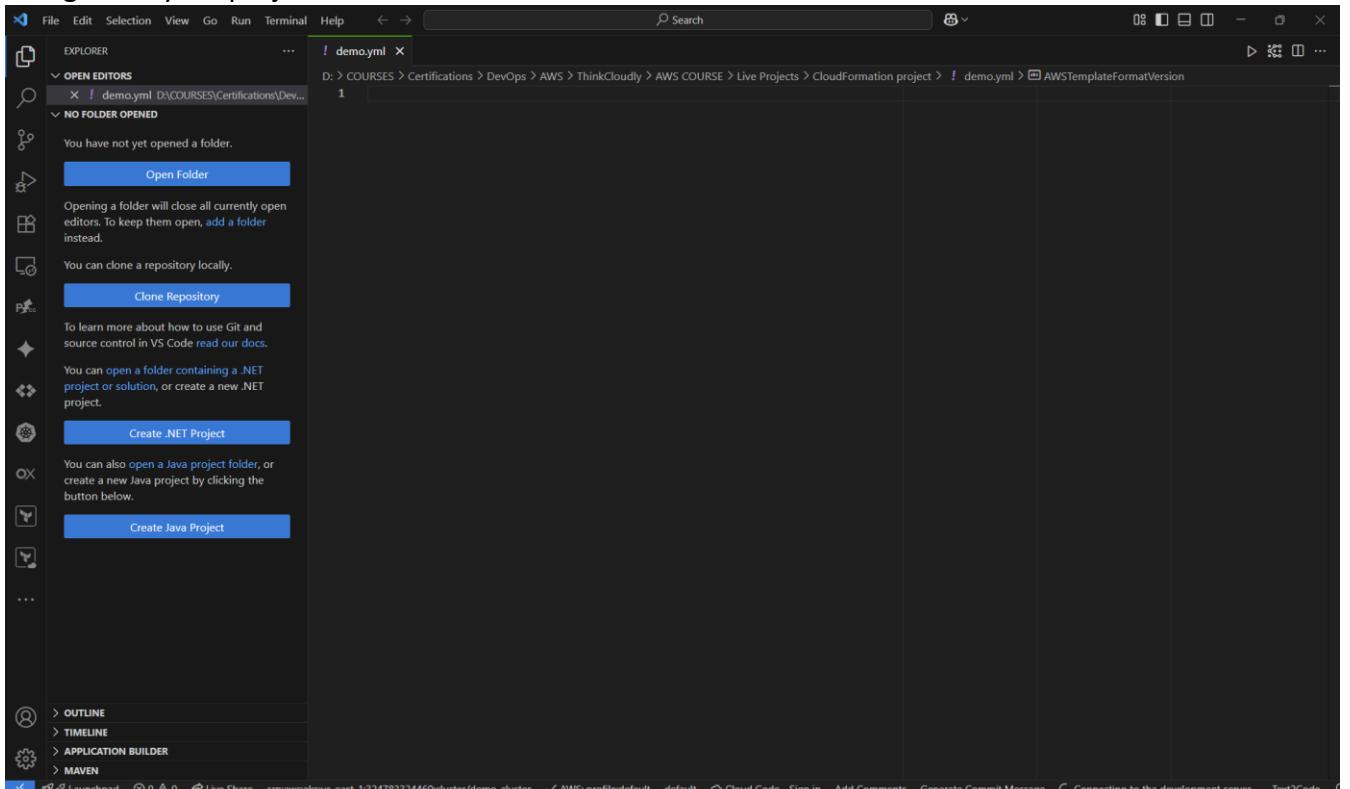


Press ENTER

Prepared by Sidney Smith Ebot

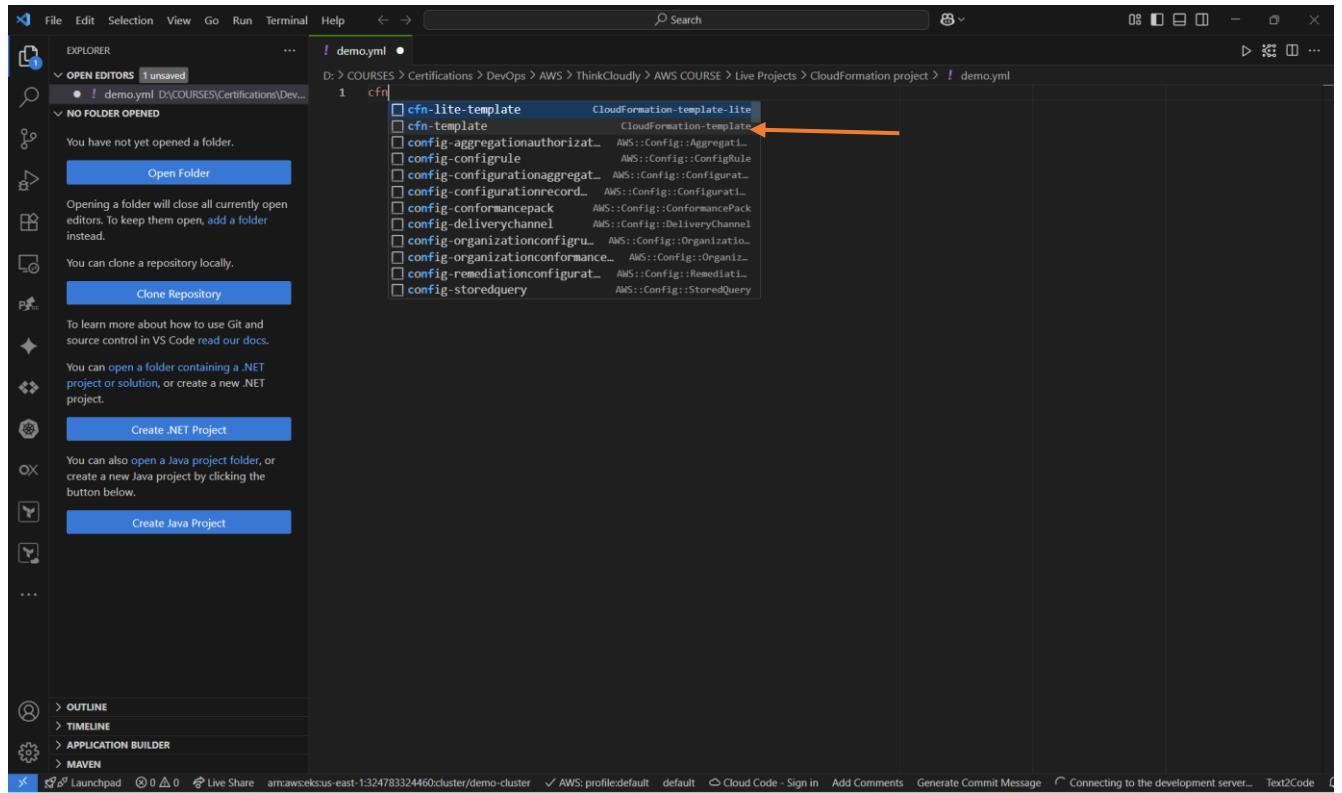


Navigate to your project folder and click on “create file”



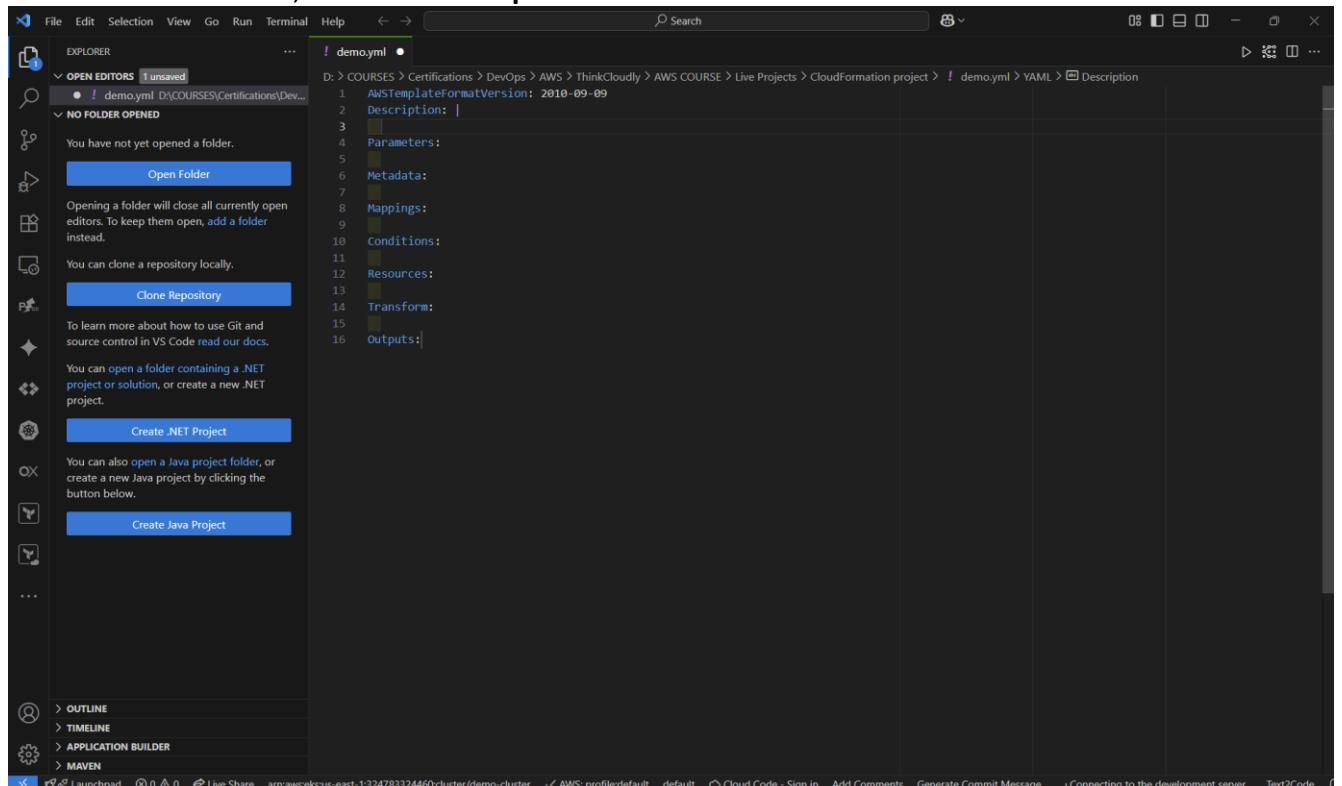
Now type “cfn”

Prepared by Sidney Smith Ebott



The screenshot shows the VS Code interface with the file `demo.yml` open. In the search bar, the prefix `cfn` is typed, and a list of suggestions appears. The suggestion `cfn-template` is highlighted with a blue box and has an orange arrow pointing to it. The list also includes other AWS-related terms like `cfn-lite-template`, `CloudFormation-template-lite`, `config-aggregationauthorizat...`, etc.

Select the second one, that is “cfn template”



The screenshot shows the VS Code interface with the file `demo.yml` open. The code editor displays the following YAML structure:

```
AWSTemplateFormatVersion: 2010-09-09
Description: |
Parameters:
Metadata:
Mappings:
Conditions:
Resources:
Transform:
Outputs:
```

Delete all the lines you do not need. I will leave only “Description” and “Resources”

Prepared by Sidney Smith Ebott

The screenshot shows the VS Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Search Bar:** Search bar at the top right.
- Explorer Panel:** Shows "OPEN EDITORS 1 unsaved" with a file named "demo.yaml". It also lists "NO FOLDER OPENED" and "You have not yet opened a folder." Below this, there are several buttons:
 - Open Folder**: Opens a folder containing all currently open editors.
 - Clone Repository**: Clones a repository locally.
 - Create .NET Project**: Creates a new .NET project.
 - Create Java Project**: Creates a new Java project.
- Bottom Status Bar:** Shows "Launchpad", "Live Share", "arm:awsseksus-east-1:324783324460:cluster/demo-cluster", "AWS:profile:default", "Cloud Code - Sign in", "Add Comments", "Generate Commit Message", "Connecting to the development server...", and "Text2Code".

On “description”, I will type “Live project cloud formation”

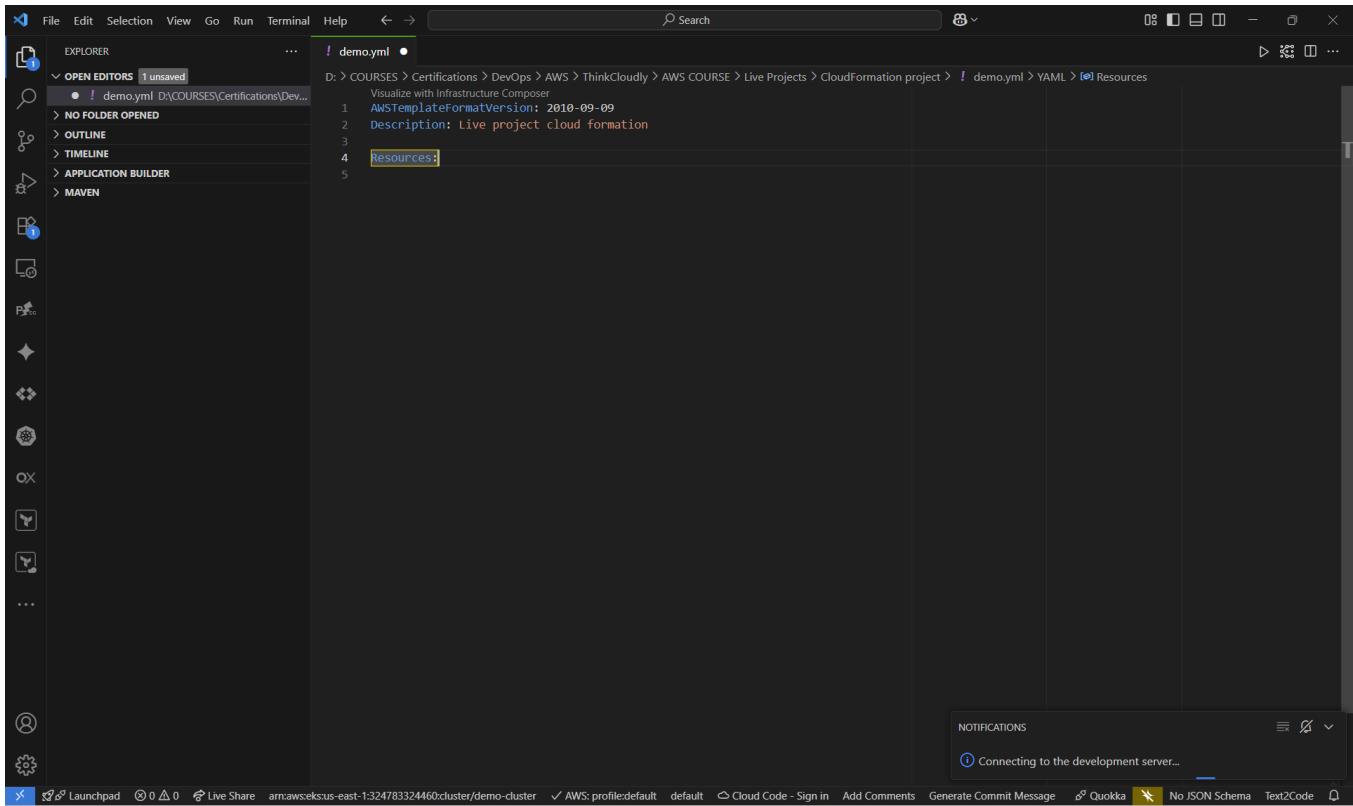
The screenshot shows the VS Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Search Bar:** Search bar at the top right.
- Explorer Panel:** Shows "OPEN EDITORS 1 unsaved" with a file named "demo.yaml". It also lists "NO FOLDER OPENED", "OUTLINE", "TIMELINE", "APPLICATION BUILDER", and "MAVEN".
- Editor Panel:** The "demo.yaml" file is open, showing the following YAML content:

```
AWS::TemplateFormatVersion: 2010-09-09
Description: Live project cloud formation
Resources:
```
- Notifications Panel:** Shows a notification: "Connecting to the development server...".
- Bottom Status Bar:** Shows "Launchpad", "Live Share", "arm:awsseksus-east-1:324783324460:cluster/demo-cluster", "AWS:profile:default", "Cloud Code - Sign in", "Add Comments", "Generate Commit Message", "Quokka", "No JSON Schema", and "Text2Code".

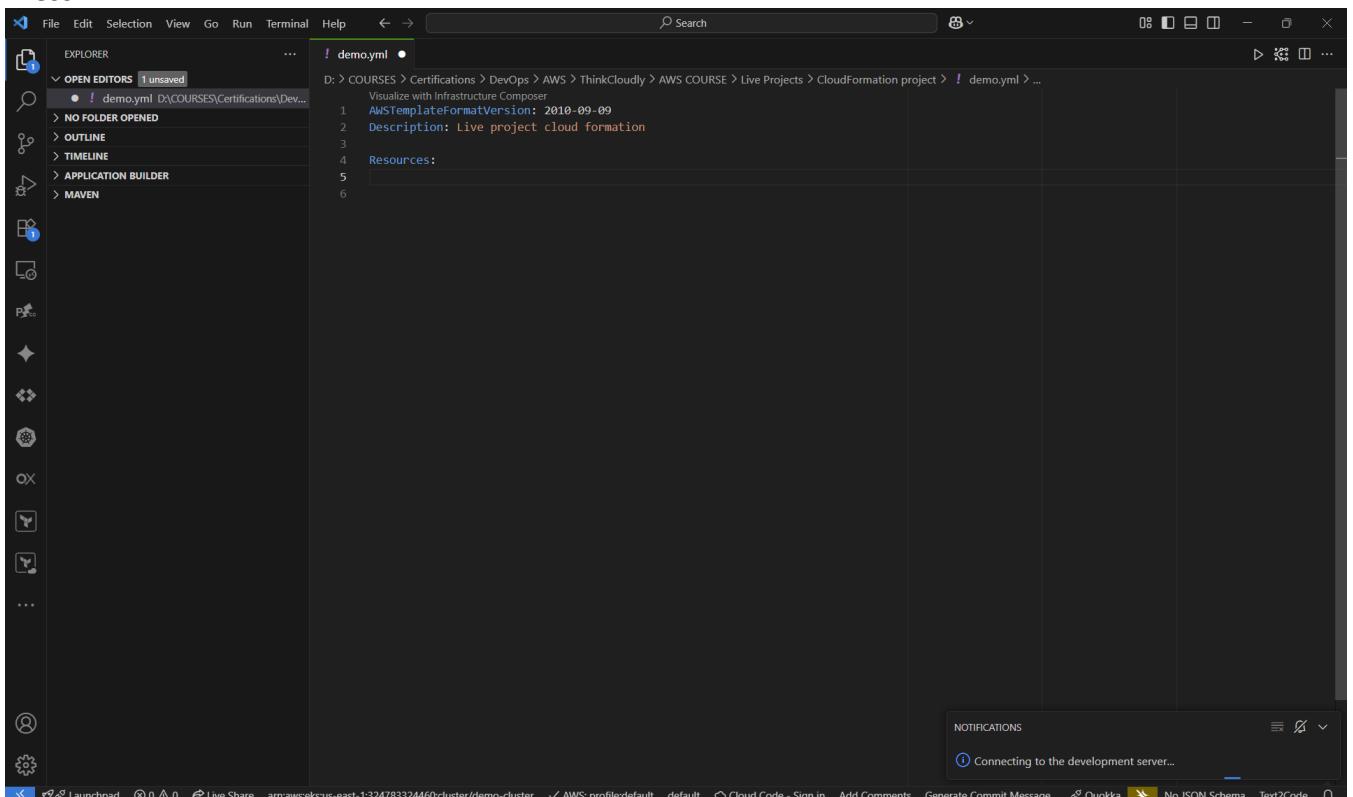
Put the cursor on the “Resources” line

Prepared by Sidney Smith Ebot



```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources  
Visualize with Infrastructure Composer  
1 AWSTemplateFormatVersion: 2010-09-09  
2 Description: Live project cloud formation  
3  
4 Resources:  
5  
6
```

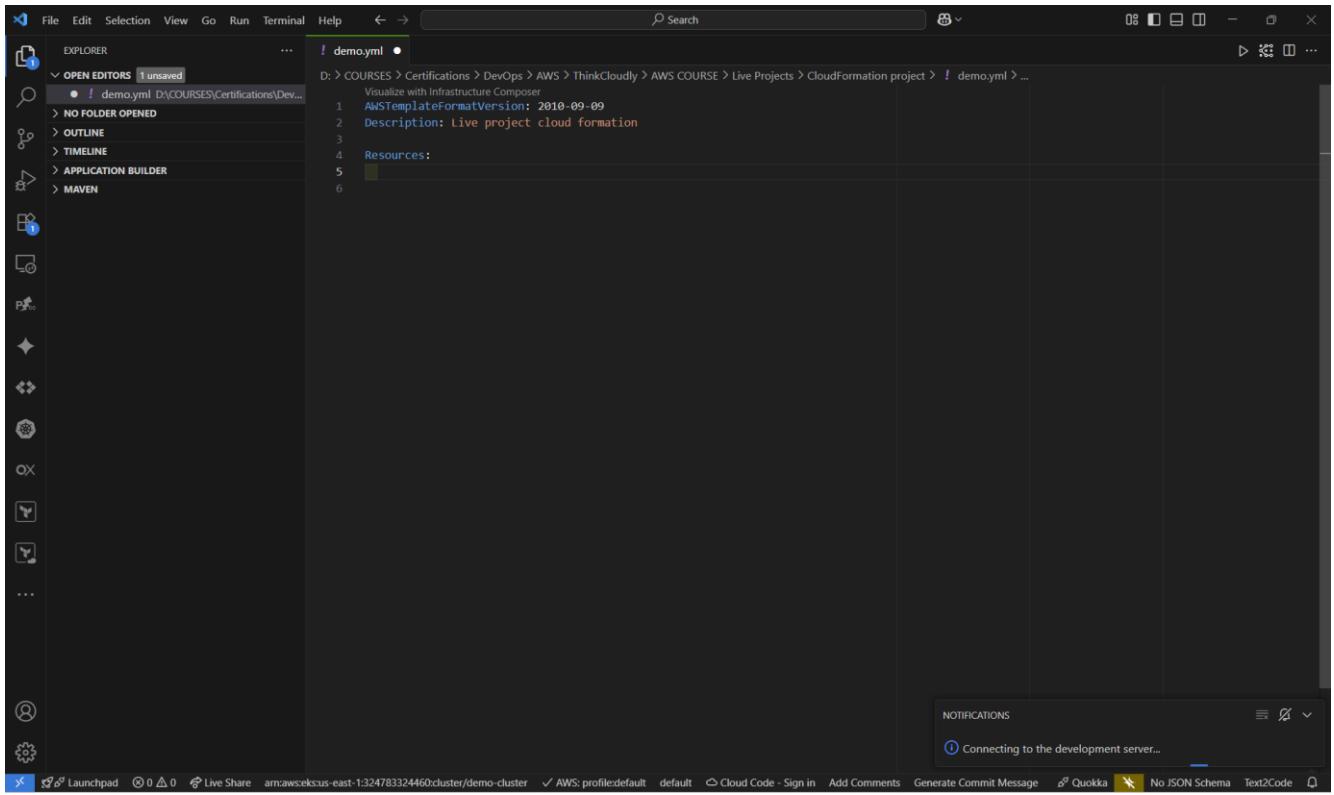
Press ENTER



```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > ...  
Visualize with Infrastructure Composer  
1 AWSTemplateFormatVersion: 2010-09-09  
2 Description: Live project cloud formation  
3  
4 Resources:  
5  
6
```

Then press “Tab” key once

Prepared by Sidney Smith Ebott

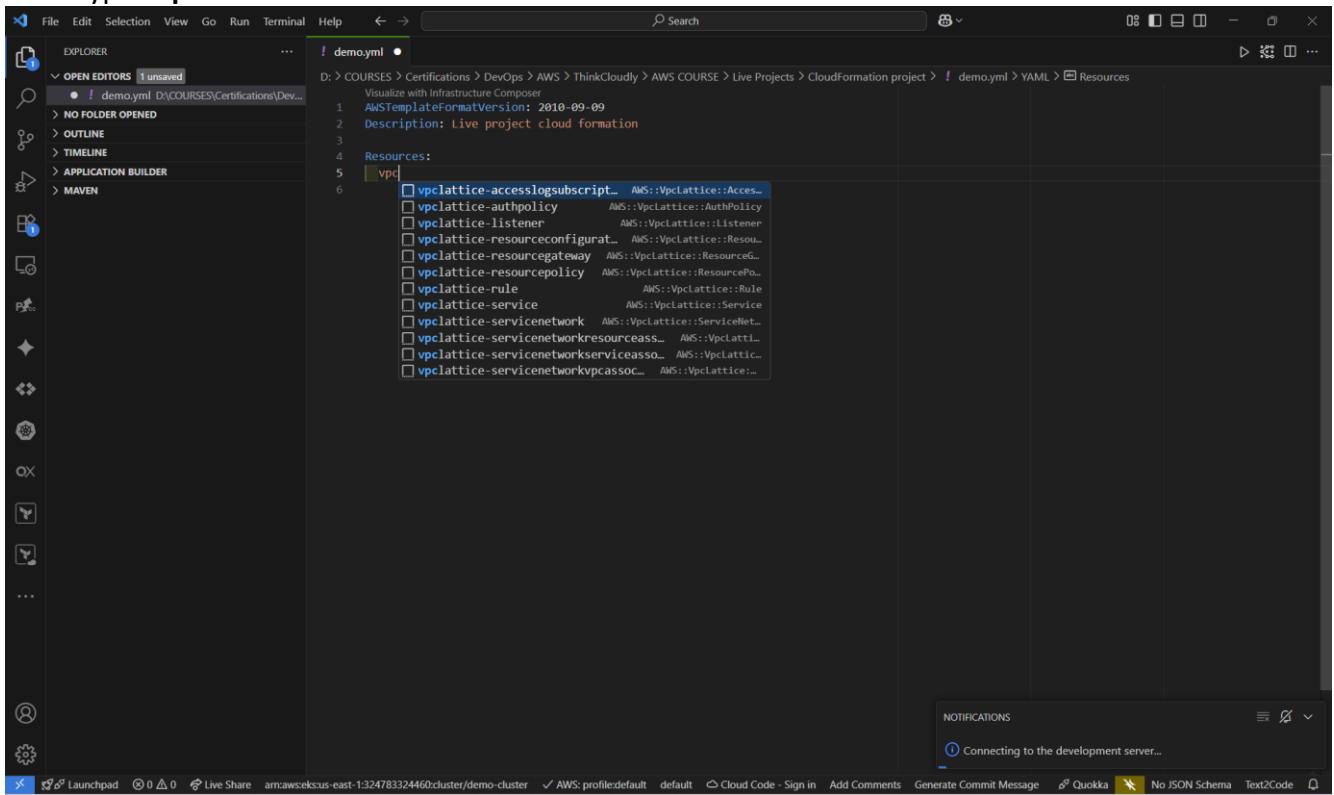


The screenshot shows a code editor interface with a dark theme. The left sidebar has icons for file operations like Open, Save, Find, and Run. The top menu includes File, Edit, Selection, View, Go, Run, Terminal, Help, and a search bar. The main editor area displays the file `demo.yaml`. The code content is as follows:

```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > ...
  1 AWSTemplateFormatVersion: 2010-09-09
  2 Description: Live project cloud formation
  3
  4 Resources:
  5
  6
```

The status bar at the bottom shows the path `arn:aws:eks:us-east-1:324783324460:cluster/demo-cluster`, profile `AWS:profiledefault`, and other development details.

Then type “vpc”

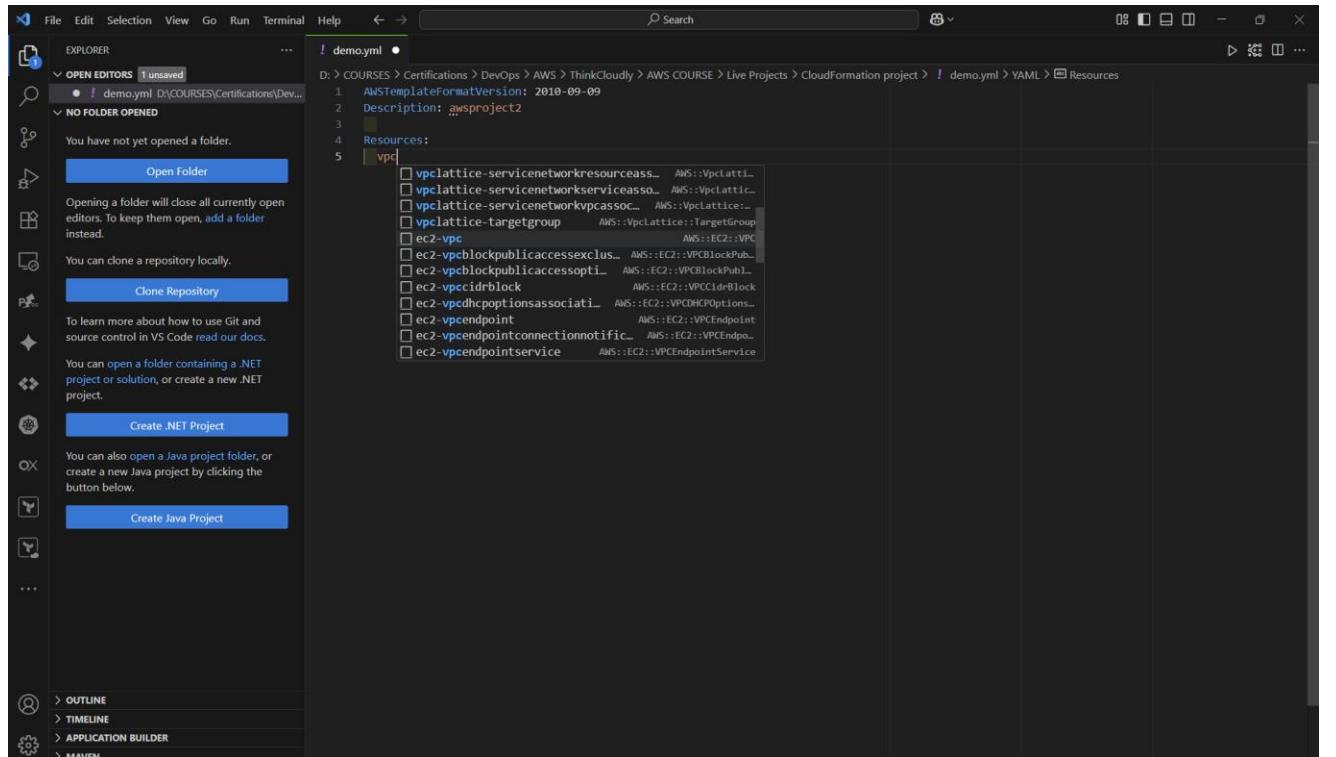


The screenshot shows the same code editor interface. After typing “vpc” in the editor, a dropdown list appears, showing a variety of AWS VPC-related resources, each preceded by a small icon and followed by its AWS namespace name.

- `vpcattice-accesslogsubscript_` AWS::VpcLattice::AccessLogSubscription
- `vpcattice-authpolicy` AWS::VpcLattice::AuthPolicy
- `vpcattice-listener` AWS::VpcLattice::Listener
- `vpcattice-resourceconfigurat...` AWS::VpcLattice::ResourceConfiguration
- `vpcattice-resourcegateway` AWS::VpcLattice::ResourceGateway
- `vpcattice-resourcepolicy` AWS::VpcLattice::ResourcePolicy
- `vpcattice-rule` AWS::VpcLattice::Rule
- `vpcattice-service` AWS::VpcLattice::Service
- `vpcattice-servicenetwork` AWS::VpcLattice::ServiceNetwork
- `vpcattice-servicenetworkresourceass...` AWS::VpcLattice::ServiceNetworkResourceAssociation
- `vpcattice-servicenetworkserviceasso...` AWS::VpcLattice::ServiceNetworkServiceAssociation
- `vpcattice-servicenetworkpcassoc...` AWS::VpcLattice::ServiceNetworkPcAssociation

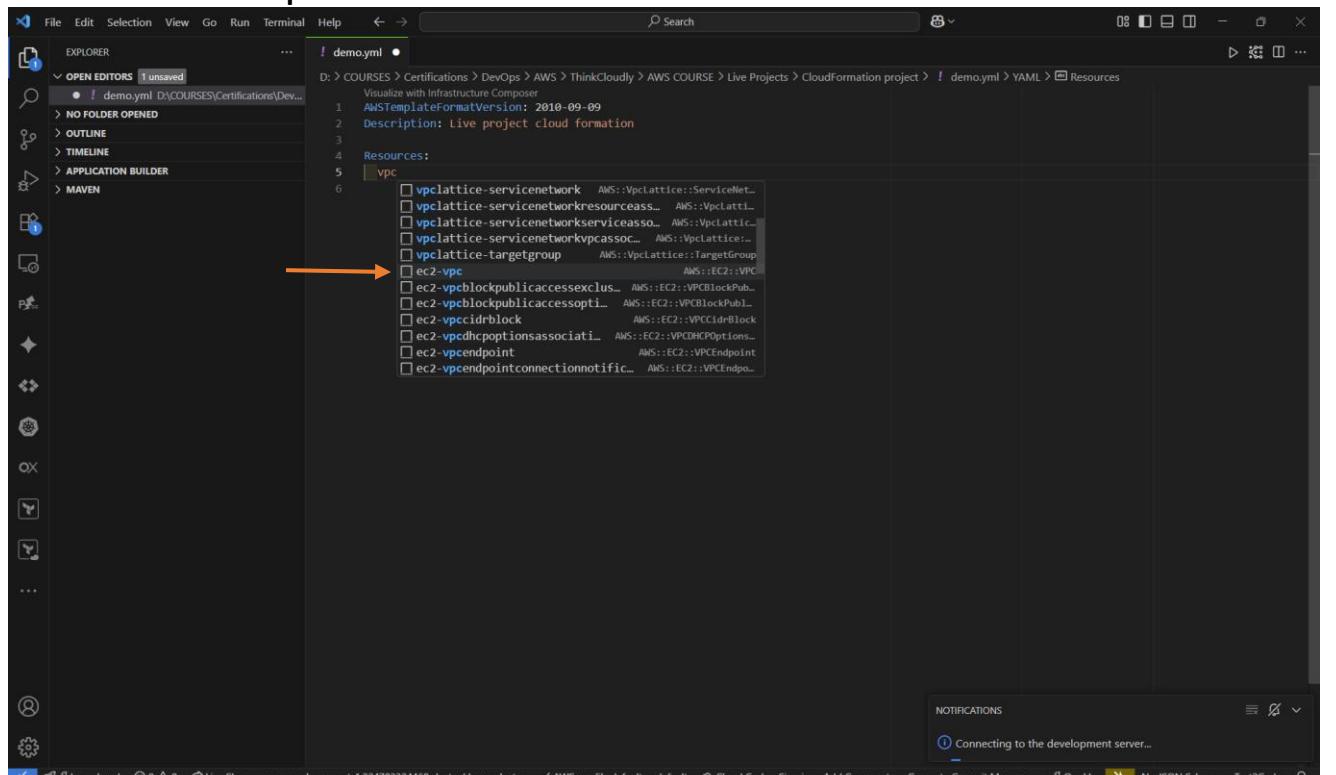
Scroll down

Prepared by Sidney Smith Ebot



```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yml > YAML > Resources
1 AWSTemplateFormatVersion: 2010-09-09
2 Description: awsproject2
3
4 Resources:
5   vpc
6     vpcLatticeServiceNetworkResourceAWS::VpcLattice::ServiceNet...
7     vpcLatticeServiceNetworkResourceAssocAWS::VpcLattice::Service...
8     vpcLatticeServiceNetworkServiceAssocAWS::VpcLattice::Service...
9     vpcLatticeTargetGroupAWS::VpcLattice::TargetGroup
10    ec2-vpc
11      AWS::EC2::VPCBlockPublicAccessExclus...
12      AWS::EC2::VPCBlockPublicAccessOpti...
13      AWS::EC2::VPCCIDRBlock
14      AWS::EC2::VPCDHCPOptionsAssociati...
15      AWS::EC2::VPCEndpoint
16      AWS::EC2::VPCEndpointConnectionNotifi...
17      AWS::EC2::VPCEndpointService
```

Scroll down to “ec2-vpc”



```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yml > YAML > Resources
1 AWSTemplateFormatVersion: 2010-09-09
2 Description: Live project cloud formation
3
4 Resources:
5   vpc
6     vpcLatticeServiceNetwork AWS::VpcLattice::ServiceNet...
7     vpcLatticeServiceNetworkResourceAWS::VpcLattice::Service...
8     vpcLatticeServiceNetworkServiceAssocAWS::VpcLattice::Service...
9     vpcLatticeTargetGroup AWS::VpcLattice::TargetGroup
10    ec2-vpc
11      AWS::EC2::VPCBlockPublicAccessExclus...
12      AWS::EC2::VPCBlockPublicAccessOpti...
13      AWS::EC2::VPCCIDRBlock
14      AWS::EC2::VPCDHCPOptionsAssociati...
15      AWS::EC2::VPCEndpoint
16      AWS::EC2::VPCEndpointConnectionNotifi...
17      AWS::EC2::VPCEndpointService
```

Select “ec2-vpc”

```

File Edit Selection View Go Run Terminal Help ← → ⌘ Search
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yml > YAML > Resources > LogicalID
OPEN EDITORS 1 unsaved
demo.yml D:\COURSES\Certifications\Dev...
NO FOLDER OPENED
OUTLINE
TIMELINE
APPLICATION BUILDER
MAVEN
Launchpad Live Share araws:eks:sus-east-1:324783324460:cluster/demo-cluster AWS:profile=default default Cloud Code - Sign in Add Comments Generate Commit Message Quokka No JSON Schema Text2Code
LogicalID:
Type: AWS::EC2::VPC
Properties:
  CidrBlock: "String"
  EnableDnsHostnames: false
  EnableDnsSupport: false
  InstanceTenancy: "String"
  Ipv4IpamPoolId: "String"
  Ipv4NetmaskLength: "Number"
Tags:
  - Key: "String"
    Value: "String"

```

Replace “logicalID” with “rdsVpc”

```

File Edit Selection View Go Run Terminal Help ← → ⌘ Search
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yml > YAML > Resources > rdsVpc > Properties > Tags
OPEN EDITORS 1 unsaved
demo.yml D:\COURSES\Certifications\Dev...
NO FOLDER OPENED
OUTLINE
TIMELINE
APPLICATION BUILDER
MAVEN
Launchpad Live Share araws:eks:sus-east-1:324783324460:cluster/demo-cluster AWS:profile=default default Cloud Code - Sign in Add Comments Generate Commit Message Quokka No JSON Schema Text2Code
rdsVpc:
Type: AWS::EC2::VPC
Properties:
  CidrBlock: "String"
  EnableDnsHostnames: false
  EnableDnsSupport: false
  InstanceTenancy: "String"
  Ipv4IpamPoolId: "String"
  Ipv4NetmaskLength: "Number"
Tags:
  - Key: "String"
    Value: "String"

```

Remove all the lines we do not need. Per our project, we need just “**CIDR Block**”, “**EnableDNSHostnames**”, and “**EnableDNSSupport**”.

The screenshot shows the AWS CloudFormation template editor in Visual Studio Code. The left sidebar displays the 'EXPLORER' view with 'OPEN EDITORS' showing '1 unsaved' file named 'demo.yml'. The main editor area contains the following CloudFormation template:

```
AWSTemplateFormatVersion: 2010-09-09
Description: Live project cloud formation

Resources:
  rdsVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: "String"
      EnableDnsHostnames: false
      EnableDnsSupport: false
      Tags:
        - Key: "String"
          Value: "String"
```

The 'NOTIFICATIONS' bar at the bottom indicates 'Connecting to the development server...'. The status bar at the bottom shows the path 'arnaws:eks:sus-east-1:324783324460:cluster/demo-cluster' and profiles 'AWS:profile/default' and 'default'.

Then enter the details:

CIDR Block: 192.168.0.0/16

EnableDNSHostnames: False

EnableDNSHostnames: False

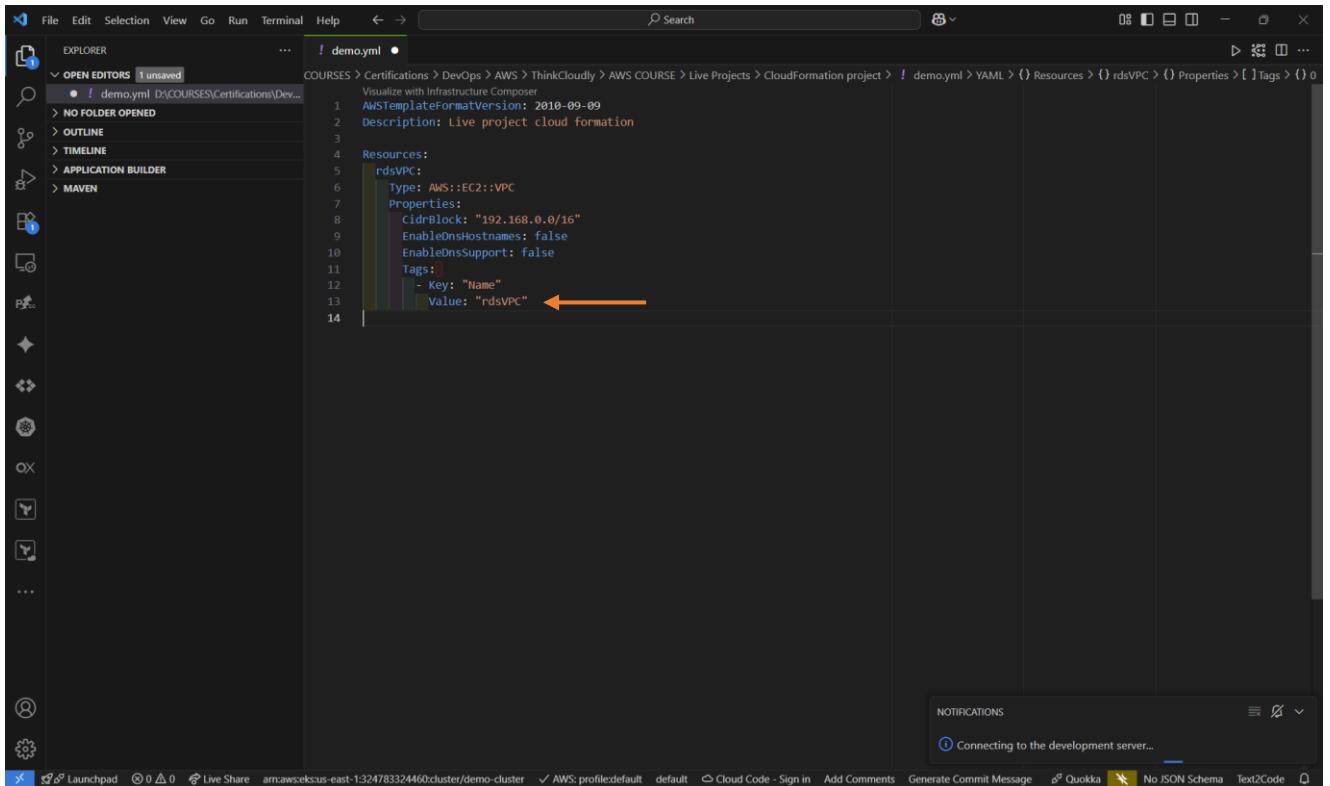
The screenshot shows the AWS CloudFormation template editor in Visual Studio Code. The 'demo.yml' file now includes the 'CidrBlock' and 'Tags' properties. An orange arrow points to the 'Key' field in the 'Tags' section, indicating where the 'Name' value should be entered.

```
AWSTemplateFormatVersion: 2010-09-09
Description: Live project cloud formation

Resources:
  rdsVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: "192.168.0.0/16"
      EnableDnsHostnames: false
      EnableDnsSupport: false
      Tags:
        - Key: "Name" ← Orange arrow points here
          Value: "rdsVPC"
```

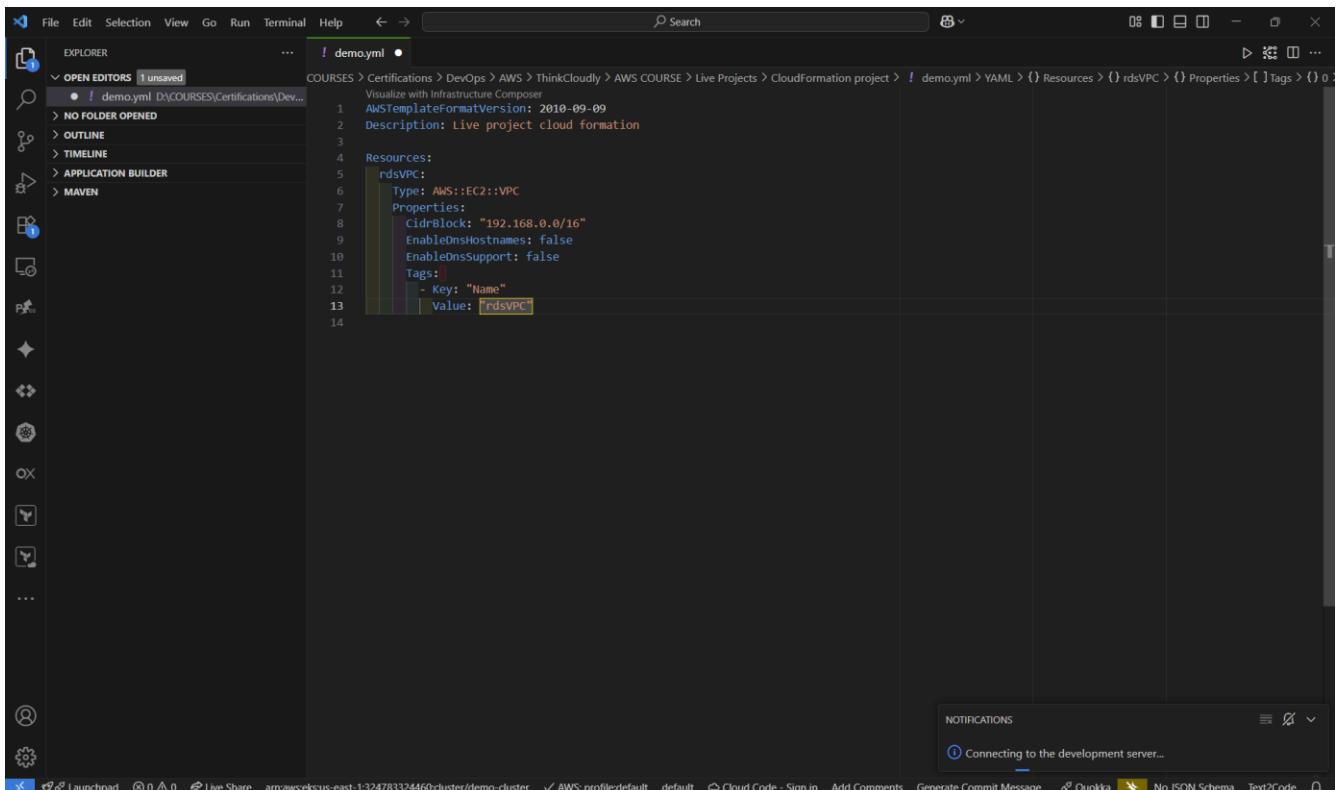
The 'NOTIFICATIONS' bar at the bottom indicates 'Connecting to the development server...'. The status bar at the bottom shows the path 'arnaws:eks:sus-east-1:324783324460:cluster/demo-cluster' and profiles 'AWS:profile/default' and 'default'.

On “key”, enter “Name” and on “value”, enter the value “rdsVPC”



```
AWS::CloudFormation::Resource
  Type: AWS::EC2::VPC
  Properties:
    CidrBlock: "192.168.0.0/16"
    EnableDnsHostnames: false
    EnableDnsSupport: false
  Tags:
    - Key: "Name"
      Value: "rdsVPC"
```

Put the cursor at the end of the value line



```
AWS::CloudFormation::Resource
  Type: AWS::EC2::VPC
  Properties:
    CidrBlock: "192.168.0.0/16"
    EnableDnsHostnames: false
    EnableDnsSupport: false
  Tags:
    - Key: "Name"
      Value: "rdsVPC"
```

Press Enter twice

Prepared by Sidney Smith Ebot

The screenshot shows the Visual Studio Code interface with the 'demo.yaml' file open in the editor. The code defines a VPC resource named 'rdsVPC' with specific properties like CIDR and tags. The 'NOTIFICATIONS' bar at the bottom indicates a connection to a development server.

```
AWSTemplateFormatVersion: 2010-09-09
Description: Live project cloud formation

Resources:
  rdsVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: "192.168.0.0/16"
      EnableDnsHostnames: false
      EnableDnsSupport: false
    Tags:
      - Key: "Name"
        Value: "rdsVPC"
```

Move the cursor backward under the line “`rds_VPC`” using the “`Backspace`” key

The screenshot shows the same AWS CloudFormation template in VS Code. The cursor is now positioned under the 'rds_VPC' line, specifically under the underscore character. The 'NOTIFICATIONS' bar continues to show the connection status.

```
AWSTemplateFormatVersion: 2010-09-09
Description: Live project cloud formation

Resources:
  rdsVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: "192.168.0.0/16"
      EnableDnsHostnames: false
      EnableDnsSupport: false
    Tags:
      - Key: "Name"
        Value: "rdsVPC"
```

Type “`subnet`”

The screenshot shows the AWS CloudFormation Visualizer interface. In the center, there is a code editor window displaying a CloudFormation YAML template named `demo.yaml`. The template defines a `rdsVPC` resource with a `subnet` property. A dropdown menu is open over the `subnet` value, listing several AWS CloudFormation resource types. An orange arrow points to the `ec2-subnet` option in the list.

```

! demo.yaml 1
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yaml > YAML > {} Resources
  1 AWS::TemplateFormatVersion: 2010-09-09
  2 Description: Live project cloud formation
  3
  4 Resources:
  5   rdsVPC:
  6     Type: AWS::EC2::VPC
  7     Properties:
  8       CidrBlock: "192.168.0.0/16"
  9       EnableDnsHostnames: false
 10      EnableDnsSupport: false
 11      Tags:
 12        - Key: "Name"
 13          Value: "rdsVPC"
 14
 15   subnet:
 16     Type: AWS::EC2::Subnet
 17     Properties:
 18       CidrBlock: "192.168.0.0/16"
 19       EnableDnsHostnames: false
 20       EnableDnsSupport: false
 21       Tags:
 22         - Key: "Name"
 23           Value: "rdsVPC"
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
  
```

Select “`ec2-subnet`”

The screenshot shows the AWS CloudFormation Visualizer interface. In the center, there is a code editor window displaying a CloudFormation YAML template named `demo.yaml`. The template defines a `rdsVPC` resource with a `subnet` property. A dropdown menu is open over the `subnet` value, listing several AWS CloudFormation resource types. An orange arrow points to the `LogicalID` field in the list.

```

! demo.yaml 1
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yaml > YAML > {} Resources > {} LogicalID
  4 Resources:
  5   rdsVPC:
  6     Properties:
  7       CidrBlock: "192.168.0.0/16"
  8       EnableDnsHostnames: false
  9       EnableDnsSupport: false
 10      Tags:
 11        - Key: "Name"
 12          Value: "rdsVPC"
 13
 14
 15   LogicalID:
 16     Type: AWS::EC2::Subnet
 17     Properties:
 18       AssignIpv6AddressOnCreation: false
 19       AvailabilityZone: "String"
 20       AvailabilityZoneId: "String"
 21       CidrBlock: "String"
 22       EnableDns64: false
 23       EnableIamAtDeviceIndex: "Number"
 24       Ipv4CidrBlock: "String"
 25       Ipv4NetmaskLength: "Number"
 26       Ipv6CidrBlock: "String"
 27       Ipv6GwipPoolId: "String"
 28       Ipv6Nat: false
 29       Ipv6NetmaskLength: "Number"
 30       MapPublicIpOnLaunch: false
 31       OutpostArn: "String"
 32       PrivateDnsNameOptionsOnLaunch:
 33         EnableResourceNameDnsAARecord: false
 34         EnableResourceNameDnsARecord: false
 35         HostnameType: "String"
 36       Tags:
 37         - Key: "String"
 38           Value: "String"
 39       VpcId: "String" # Required
 40
  
```

On “`logicalID`”, enter “`PrivateSubnet1`”

```

Resources:
  rdsVPC:
    Properties:
      CidrBlock: "192.168.0.0/16"
      EnableDnsHostnames: false
      EnableDnsSupport: false
      Tags:
        - Key: "Name"
          Value: "rdsVPC"

PrivateSubnet:
  Type: AWS::EC2::Subnet
  Properties:
    AssignIpv6AddressOnCreation: false
    AvailabilityZone: "String"
    AvailabilityZoneId: "String"
    CidrBlock: "String"
    EnableDns4: false
    EnableInitDeviceIndex: "Number"
    Ipv4IpamPoolId: "String"
    Ipv4NetmaskLength: "Number"
    Ipv6CidrBlock: "String"
    Ipv6IpamPoolId: "String"
    Ipv6Native: false
    Ipv6NetmaskLength: "Number"
    MapPublicIpOnLaunch: false
    OutpostArm: "String"
    PrivateDnsNameOptionsOnLaunch:
      EnableResourceNameDnsAAAARecord: false
      EnableResourceNameDnsARecord: false
      HostnameType: "String"
    Tags:
      - Key: "String"
        Value: "String"
    VpcId: "String" # Required
  
```

Remove the lines that are not needed.

```

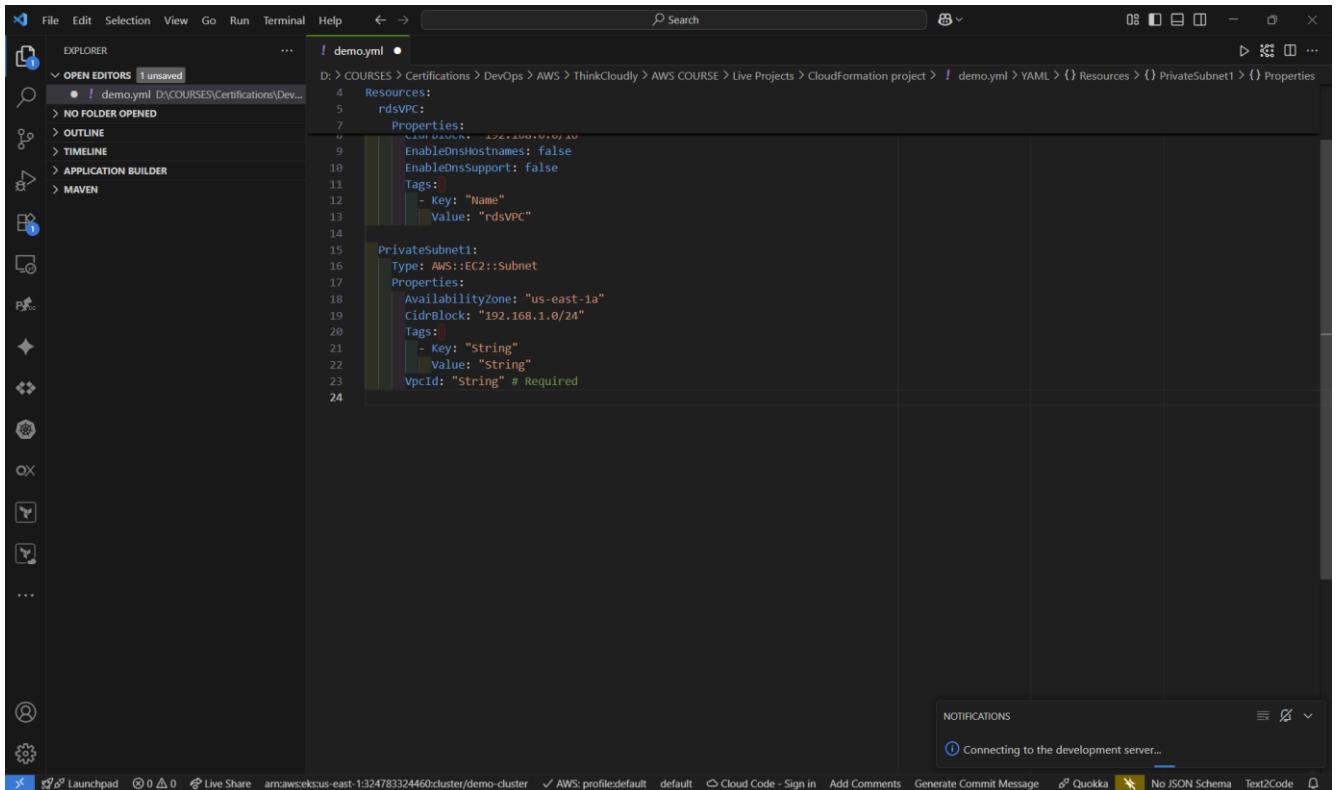
Resources:
  rdsVPC:
    Properties:
      CidrBlock: "192.168.0.0/16"
      EnableDnsHostnames: false
      EnableDnsSupport: false
      Tags:
        - Key: "Name"
          Value: "rdsVPC"

PrivateSubnet:
  Type: AWS::EC2::Subnet
  VpcId: "String" # Required
  
```

Enter these details:

AvailabilityZone: us-east-1a

CidrBlock: 192.168.1.0/24

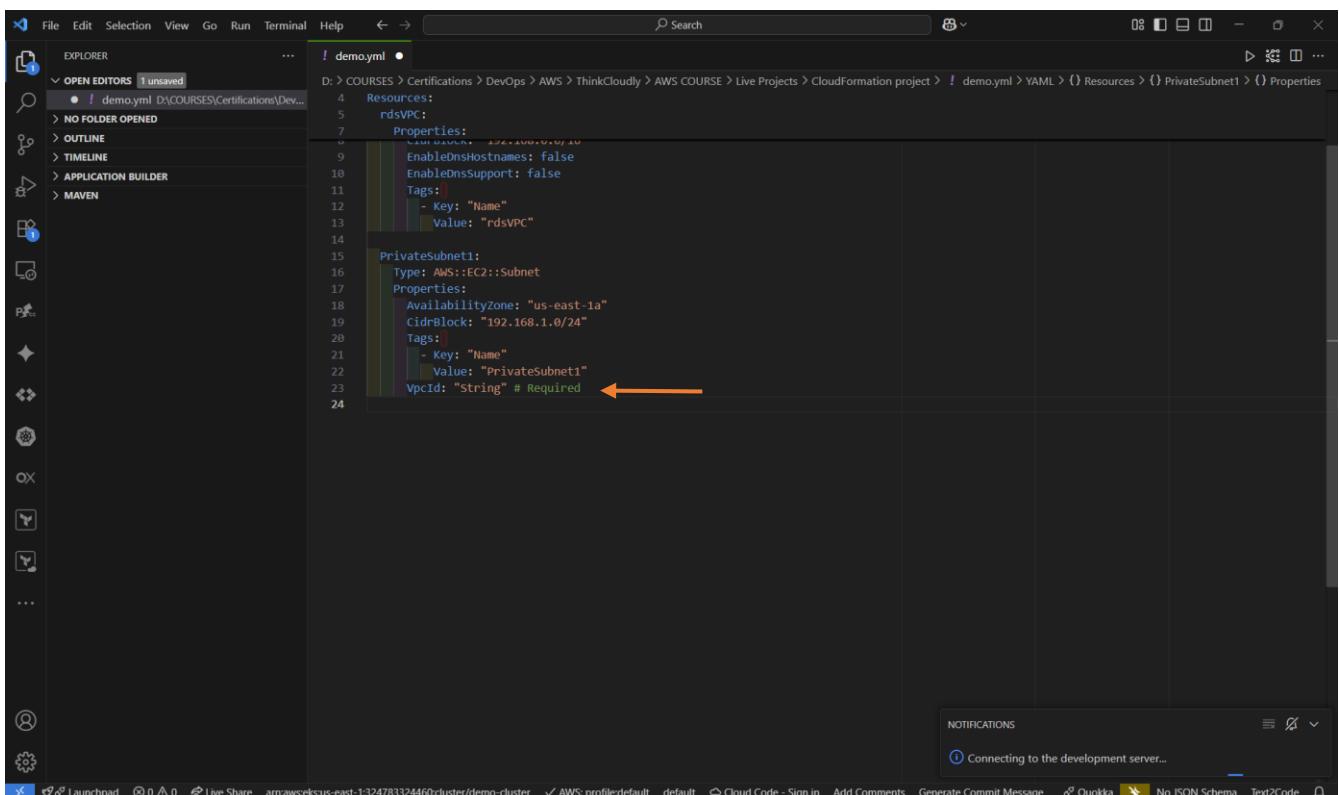


```

1 Resources:
2   rdsVPC:
3     Properties:
4       CidrBlock: 192.168.0.0/10
5       EnableDnsHostnames: false
6       EnableDnsSupport: false
7       Tags:
8         - Key: "Name"
9           Value: "rdsVPC"
10
11 PrivateSubnet1:
12   Type: AWS::EC2::Subnet
13   Properties:
14     AvailabilityZone: "us-east-1a"
15     CidrBlock: "192.168.1.0/24"
16     Tags:
17       - Key: "String"
18         Value: "String"
19     VpcId: "String" # Required
20
21
22
23
24

```

Under “Tag”, in “key” enter “Name” and in “value” enter “PrivateSubnet1”



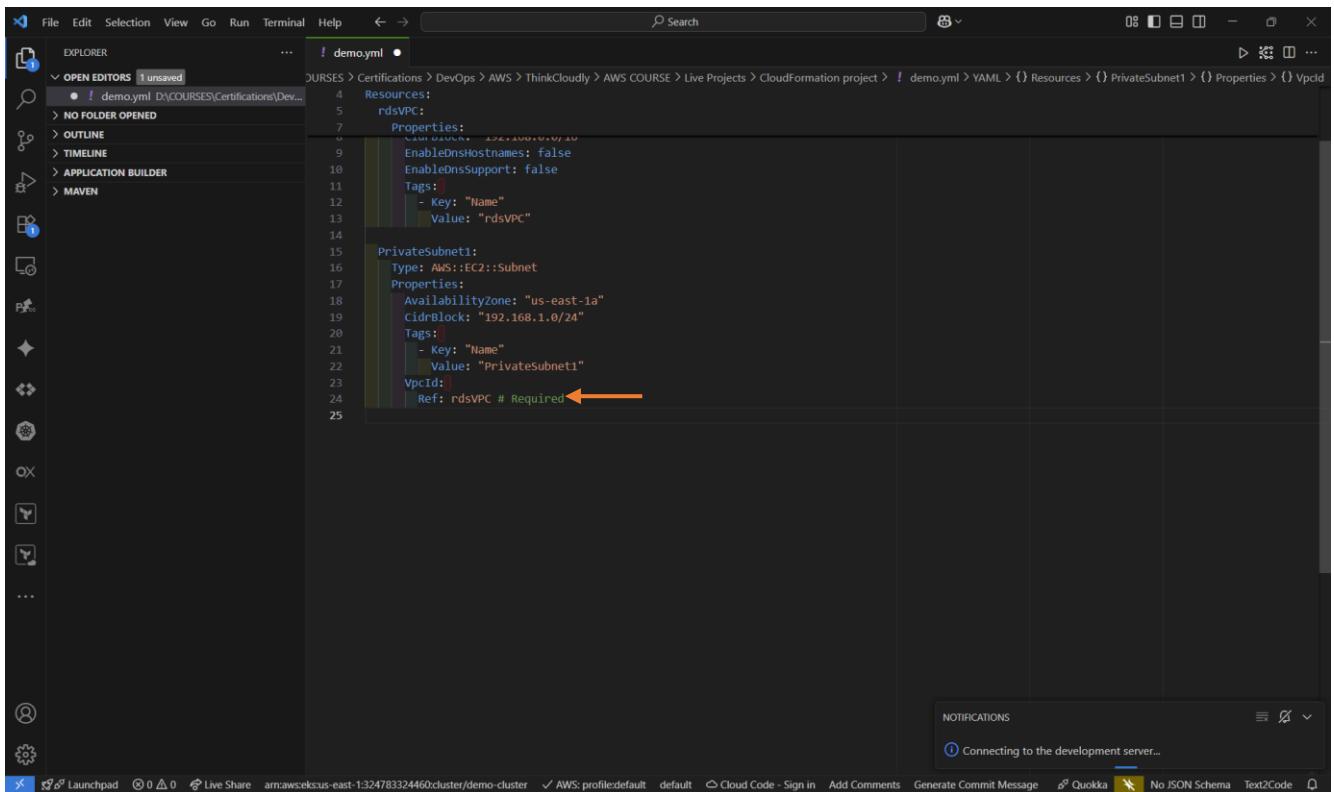
```

1 Resources:
2   rdsVPC:
3     Properties:
4       CidrBlock: 192.168.0.0/10
5       EnableDnsHostnames: false
6       EnableDnsSupport: false
7       Tags:
8         - Key: "Name"
9           Value: "rdsVPC"
10
11 PrivateSubnet1:
12   Type: AWS::EC2::Subnet
13   Properties:
14     AvailabilityZone: "us-east-1a"
15     CidrBlock: "192.168.1.0/24"
16     Tags:
17       - Key: "Name"
18         Value: "PrivateSubnet1"
19     VpcId: "String" # Required
20
21
22
23
24

```

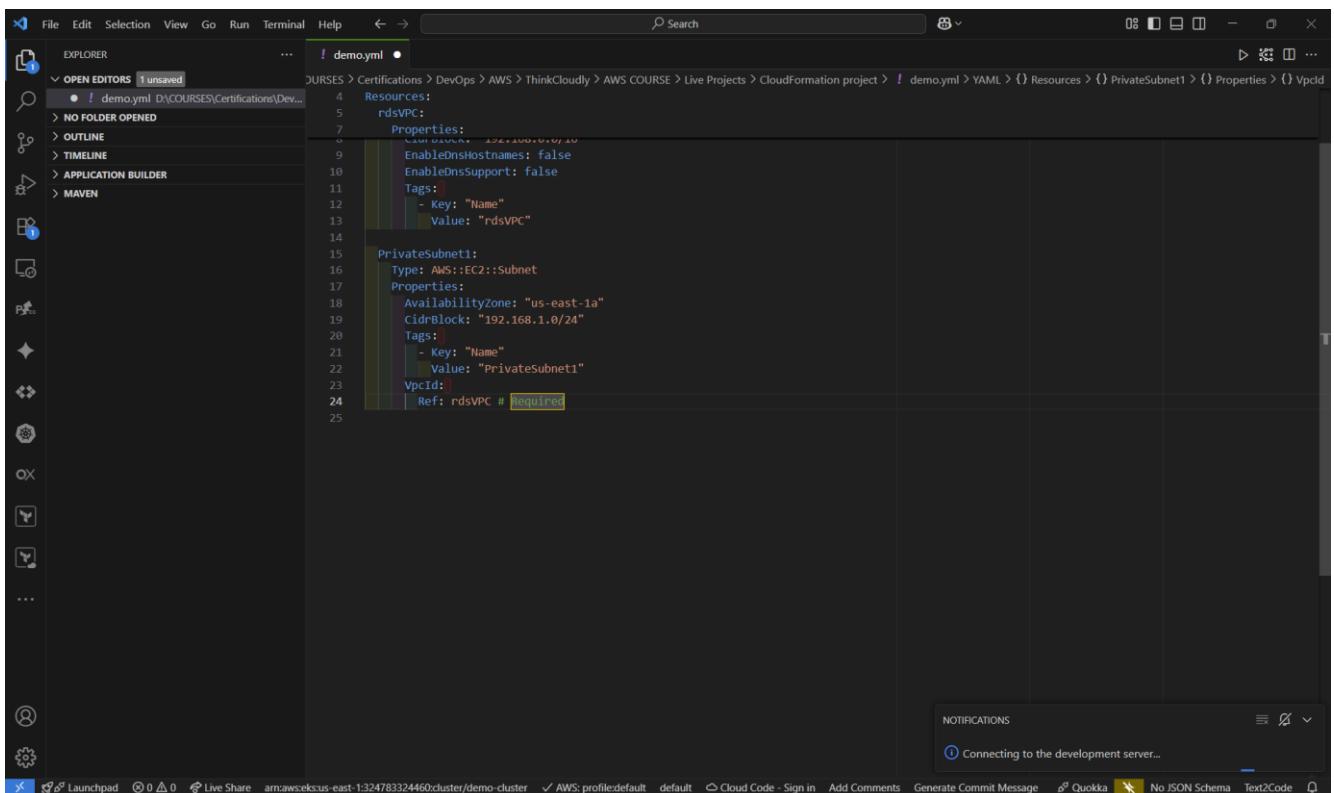
On “vpcId” enter “Ref: rdsVpc”. The Ref value (**rdsVPC**) should match with the name of the VPC (**rdsVPC**)

Prepared by Sidney Smith Ebott



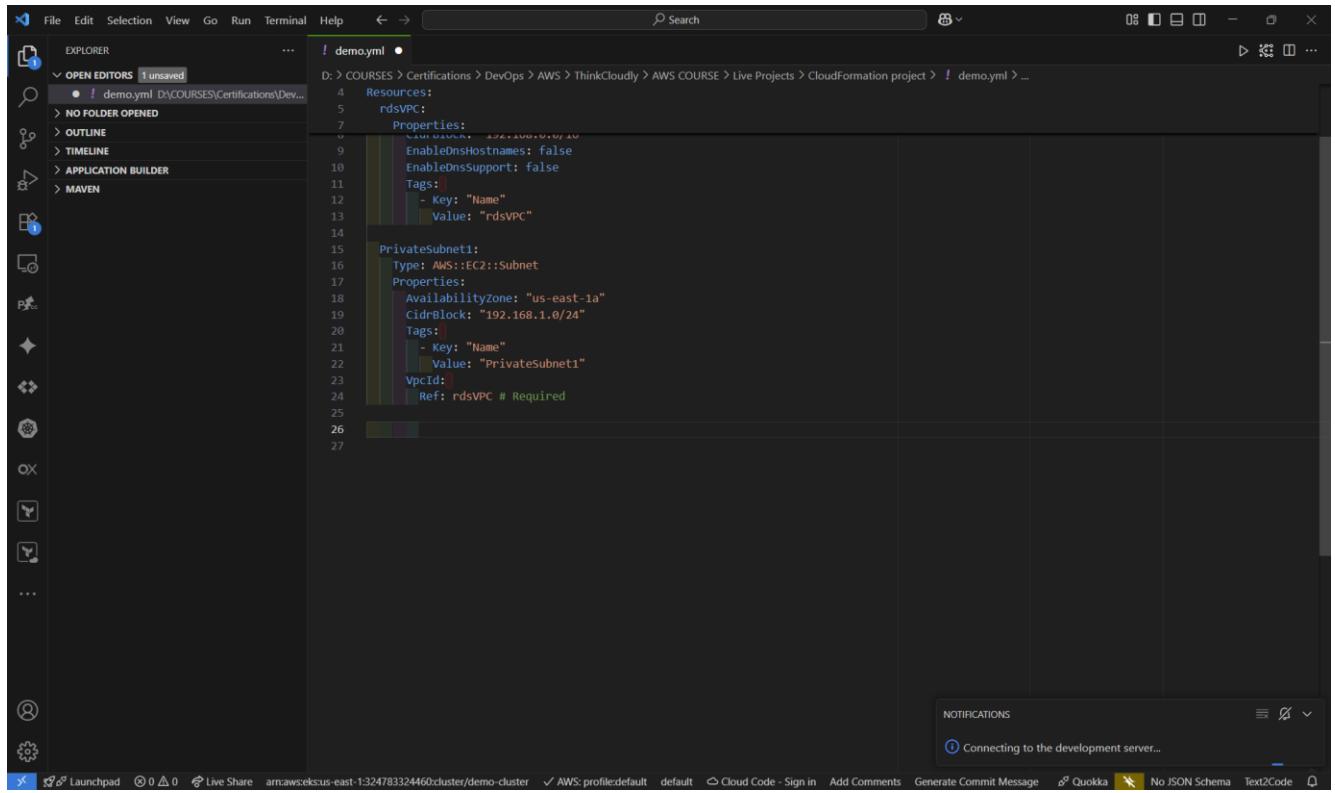
```
demo.yaml
COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > PrivateSubnet1 > Properties > VpcId
1 Resources:
2   rdsVPC:
3     Properties:
4       CidrBlock: "192.168.0.0/16"
5       EnableDnsHostnames: false
6       EnableDnsSupport: false
7       Tags:
8         - Key: "Name"
9           Value: "rdsVPC"
10
11 PrivateSubnet1:
12   Type: AWS::EC2::Subnet
13   Properties:
14     AvailabilityZone: "us-east-1a"
15     CidrBlock: "192.168.1.0/24"
16     Tags:
17       - Key: "Name"
18         Value: "PrivateSubnet1"
19     VpcId: Ref: rdsVPC # Required
```

Put your cursor at the end of the “`vpcId`” line



```
demo.yaml
COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > PrivateSubnet1 > Properties > VpcId
1 Resources:
2   rdsVPC:
3     Properties:
4       CidrBlock: "192.168.0.0/16"
5       EnableDnsHostnames: false
6       EnableDnsSupport: false
7       Tags:
8         - Key: "Name"
9           Value: "rdsVPC"
10
11 PrivateSubnet1:
12   Type: AWS::EC2::Subnet
13   Properties:
14     AvailabilityZone: "us-east-1a"
15     CidrBlock: "192.168.1.0/24"
16     Tags:
17       - Key: "Name"
18         Value: "PrivateSubnet1"
19     VpcId: Ref: rdsVPC # required
```

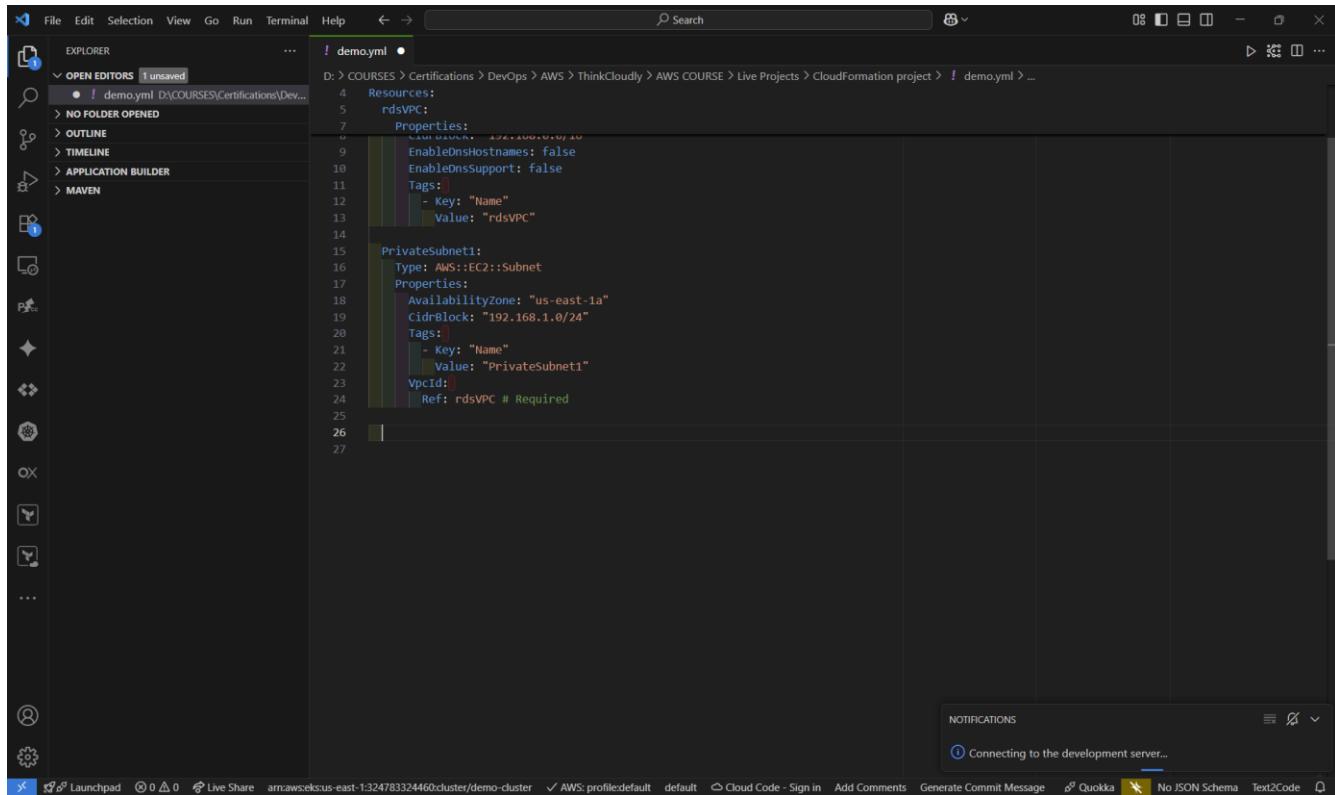
Press ENTER twice



The screenshot shows the VS Code interface with the 'demo.yaml' file open in the editor. The cursor is positioned on the line 'PrivateSubnet1:' at line 15. The code editor has a dark theme. The 'NOTIFICATIONS' panel on the right shows a message: 'Connecting to the development server...'. The bottom status bar indicates the file path as 'D:\COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml'.

```
demo.yaml
4 Resources:
5   rdsVPC:
6     Properties:
7       CidrBlock: 192.168.0.0/10
8       EnableDnsHostnames: false
9       EnableDnsSupport: false
10      Tags:
11        - Key: "Name"
12          Value: "rdsVPC"
13
14 PrivateSubnet1:
15   Type: AWS::EC2::Subnet
16   Properties:
17     AvailabilityZone: "us-east-1a"
18     CidrBlock: "192.168.1.0/24"
19     Tags:
20       - Key: "Name"
21         Value: "PrivateSubnet1"
22     VpcId:
23       Ref: rdsVPC # Required
24
25
26
27
```

Then move the cursor backward to the line “PrivateSubnet1”



The screenshot shows the VS Code interface with the 'demo.yaml' file open in the editor. The cursor is now positioned on the line 'PrivateSubnet1:' at line 15. The code editor has a dark theme. The 'NOTIFICATIONS' panel on the right shows a message: 'Connecting to the development server...'. The bottom status bar indicates the file path as 'D:\COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml'.

```
demo.yaml
4 Resources:
5   rdsVPC:
6     Properties:
7       CidrBlock: 192.168.0.0/10
8       EnableDnsHostnames: false
9       EnableDnsSupport: false
10      Tags:
11        - Key: "Name"
12          Value: "rdsVPC"
13
14 PrivateSubnet1:
15   Type: AWS::EC2::Subnet
16   Properties:
17     AvailabilityZone: "us-east-1a"
18     CidrBlock: "192.168.1.0/24"
19     Tags:
20       - Key: "Name"
21         Value: "PrivateSubnet1"
22     VpcId:
23       Ref: rdsVPC # Required
24
25
26
27
```

Type “subnet”

```

1 Resources:
2   rdsVPC:
3     Properties:
4       CidrBlock: "192.168.0.0/10"
5       EnableDnsHostnames: false
6       EnableDnsSupport: false
7       Tags:
8         - Key: "Name"
9           Value: "rdsVPC"
10
11      PrivateSubnet1:
12        Type: AWS::EC2::Subnet
13        Properties:
14          AvailabilityZone: "us-east-1a"
15          CidrBlock: "192.168.1.0/24"
16          Tags:
17            - Key: "Name"
18              Value: "PrivateSubnet1"
19          VpcId:
20            Ref: rdsVPC # Required
21
22      subnet:
23        Type: AWS::EC2::Subnet
24        Properties:
25          CidrBlock: "192.168.1.0/24"
26          Tags:
27            - Key: "Name"
28              Value: "PrivateSubnet1"
29          VpcId:
30            Ref: rdsVPC # Required
31
32      logicalID:
33        Type: AWS::EC2::Subnet
34        Properties:
35          AssignIpv6AddressOnCreation: false
36          AvailabilityZone: "String"
37          AvailabilityZoneId: "String"
38          CidrBlock: "String"
39          EnableDns4: false
40          EnableIpv6AtDeviceIndex: "Number"
41          Ipv4IpamPoolId: "String"
42          Ipv4NetmaskLength: "Number"
43          Ipv6CidrBlock: "String"
44          Ipv6IpamPoolId: "String"
45          Ipv6Native: false
46          Ipv6NetmaskLength: "Number"
47          MapPublicIpOnLaunch: false
48          OutpostArn: "String"
49          PrivateDnsNameOptionsOnLaunch:
50            EnableResourceNameDnsAAAARecord: false
51            EnableResourceNameDnsSRRecord: false
52            HostnameType: "String"
53            Tags:
54              - Key: "String"
55                Value: "String"
56            VpcId: "String" # Required

```

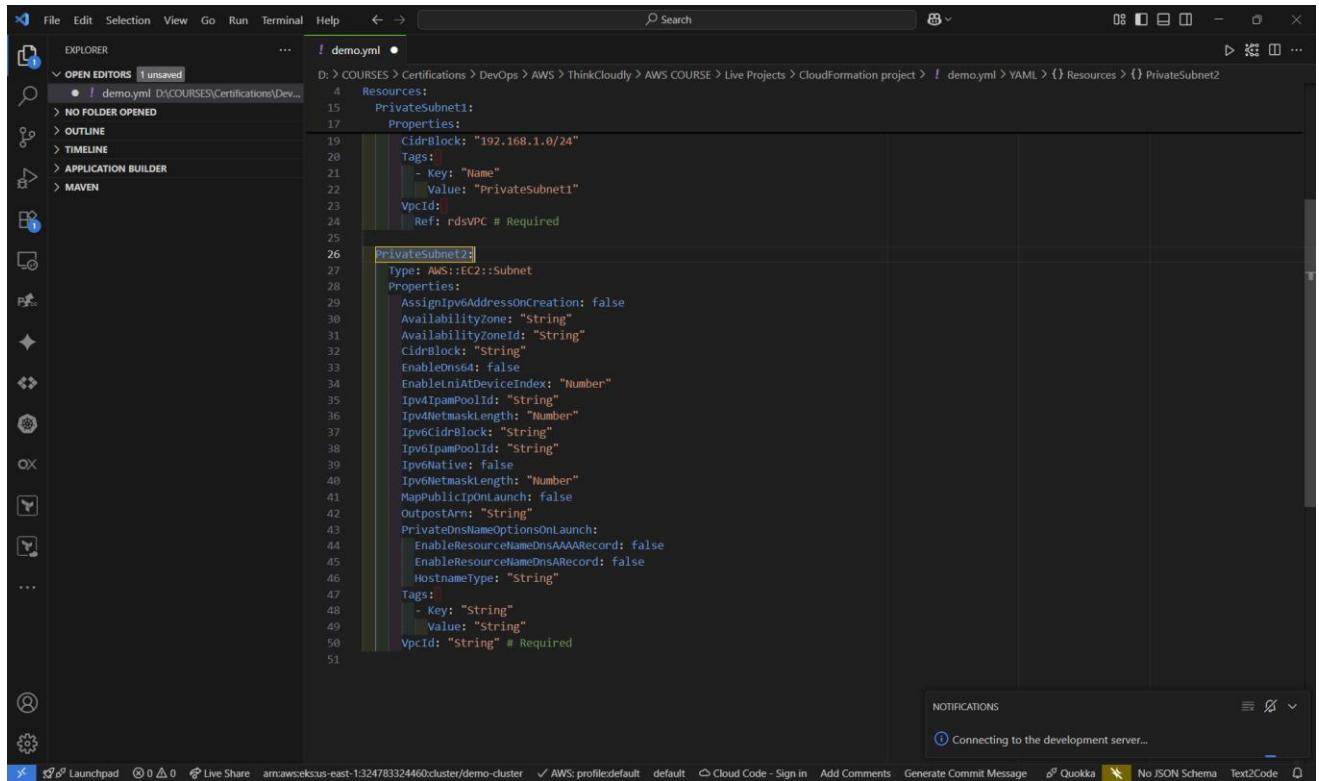
Select “ec2-subnet”

```

1 Resources:
2   rdsVPC:
3     Properties:
4       CidrBlock: "192.168.0.0/10"
5       EnableDnsHostnames: false
6       EnableDnsSupport: false
7       Tags:
8         - Key: "Name"
9           Value: "rdsVPC"
10
11      PrivateSubnet1:
12        Type: AWS::EC2::Subnet
13        Properties:
14          AvailabilityZone: "us-east-1a"
15          CidrBlock: "192.168.1.0/24"
16          Tags:
17            - Key: "Name"
18              Value: "PrivateSubnet1"
19          VpcId:
20            Ref: rdsVPC # Required
21
22      subnet:
23        Type: AWS::EC2::Subnet
24        Properties:
25          CidrBlock: "192.168.1.0/24"
26          Tags:
27            - Key: "Name"
28              Value: "PrivateSubnet1"
29          VpcId:
30            Ref: rdsVPC # Required
31
32      logicalID:
33        Type: AWS::EC2::Subnet
34        Properties:
35          AssignIpv6AddressOnCreation: false
36          AvailabilityZone: "String"
37          AvailabilityZoneId: "String"
38          CidrBlock: "String"
39          EnableDns4: false
40          EnableIpv6AtDeviceIndex: "Number"
41          Ipv4IpamPoolId: "String"
42          Ipv4NetmaskLength: "Number"
43          Ipv6CidrBlock: "String"
44          Ipv6IpamPoolId: "String"
45          Ipv6Native: false
46          Ipv6NetmaskLength: "Number"
47          MapPublicIpOnLaunch: false
48          OutpostArn: "String"
49          PrivateDnsNameOptionsOnLaunch:
50            EnableResourceNameDnsAAAARecord: false
51            EnableResourceNameDnsSRRecord: false
52            HostnameType: "String"
53            Tags:
54              - Key: "String"
55                Value: "String"
56            VpcId: "String" # Required

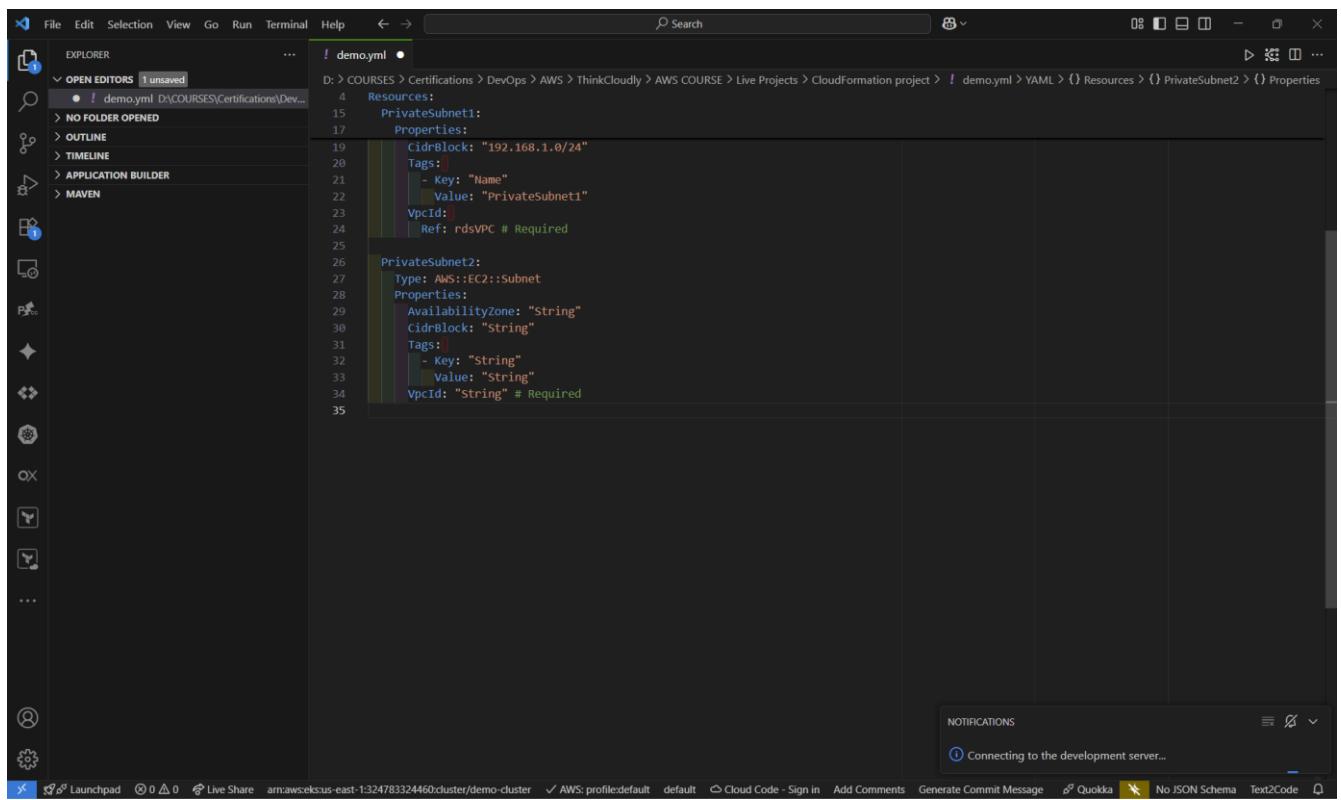
```

Replace “logicalID” with the name “PrivateSubnet2”



```
demo.yaml
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > PrivateSubnet2
  Resources:
    PrivateSubnet1:
      Properties:
        CidrBlock: "192.168.1.0/24"
        Tags:
          - Key: "Name"
            Value: "PrivateSubnet1"
        VpcId:
          Ref: rdsVPC # Required
    PrivateSubnet2:
      Type: AWS::EC2::Subnet
      Properties:
        AssignIpv6AddressOnCreation: false
        AvailabilityZone: "String"
        AvailabilityZoneId: "String"
        CidrBlock: "String"
        EnableDns64: false
        EnableIamInstanceProfile: false
        Ipv4IpamPoolId: "String"
        Ipv4NetmaskLength: "Number"
        MapPublicIpOnLaunch: false
        OutpostArn: "String"
        PrivateDnsNameOptionsOnLaunch:
          EnableResourceNameDnsAAAARecord: false
          EnableResourceNameDnsARecord: false
        HostnameType: "String"
        Tags:
          - Key: "String"
            Value: "String"
        VpcId: "String" # Required
```

Remove the lines that we do not need



```
demo.yaml
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > PrivateSubnet2 > Properties
  Resources:
    PrivateSubnet1:
      Properties:
        CidrBlock: "192.168.1.0/24"
        Tags:
          - Key: "Name"
            Value: "PrivateSubnet1"
        VpcId:
          Ref: rdsVPC # Required
    PrivateSubnet2:
      Type: AWS::EC2::Subnet
      Properties:
        AvailabilityZone: "string"
        CidrBlock: "String"
        Tags:
          - Key: "String"
            Value: "String"
        VpcId: "String" # Required
```

Enter these details:

CidrBlock: us-east-1b

AvailabilityZone: 192.168.2.0/24

```

1 demo.yml •
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > PrivateSubnet2 > Properties
4 Resources:
15   PrivateSubnet1:
16     Properties:
17       CidrBlock: "192.168.1.0/24"
18       Tags:
19         - Key: "Name"
20           Value: "PrivateSubnet1"
21       VpcId:
22         Ref: rdsVPC # Required
23
24
25   PrivateSubnet2:
26     Type: AWS::EC2::Subnet
27     Properties:
28       AvailabilityZone: "us-east-1b"
29       CidrBlock: "192.168.2.0/24"
30       Tags:
31         - Key: "Name"
32           Value: "String"
33       VpcId: "String" # Required
34
35

```

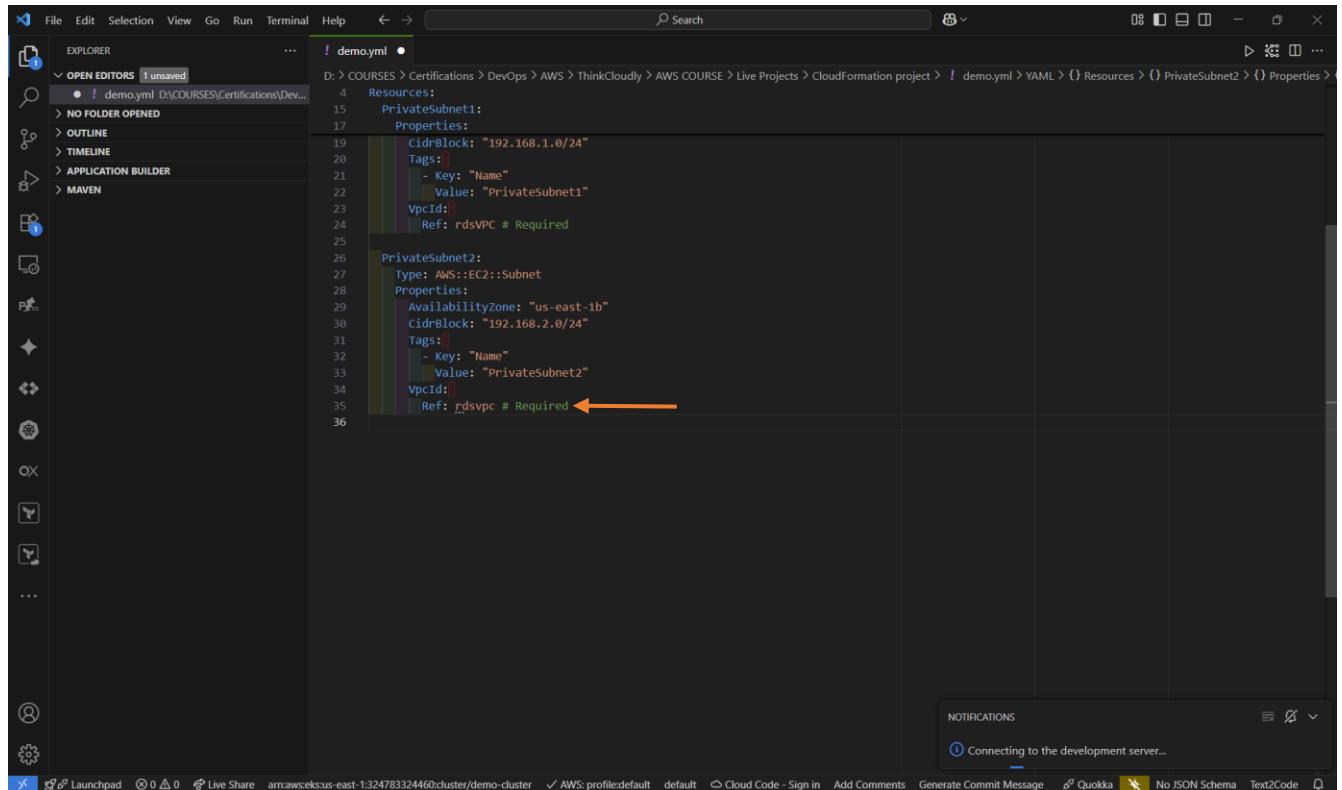
Under “Tag”, in “key” enter “Name” and in “value” enter “PrivateSubnet2”

```

1 demo.yml •
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > PrivateSubnet2 > Properties
4 Resources:
15   PrivateSubnet1:
16     Properties:
17       CidrBlock: "192.168.1.0/24"
18       Tags:
19         - Key: "Name"
20           Value: "PrivateSubnet1"
21       VpcId:
22         Ref: rdsVPC # Required
23
24
25   PrivateSubnet2:
26     Type: AWS::EC2::Subnet
27     Properties:
28       AvailabilityZone: "us-east-1b"
29       CidrBlock: "192.168.2.0/24"
30       Tags:
31         - Key: "Name"
32           Value: "String" ←
33       VpcId: "String" # Required
34
35

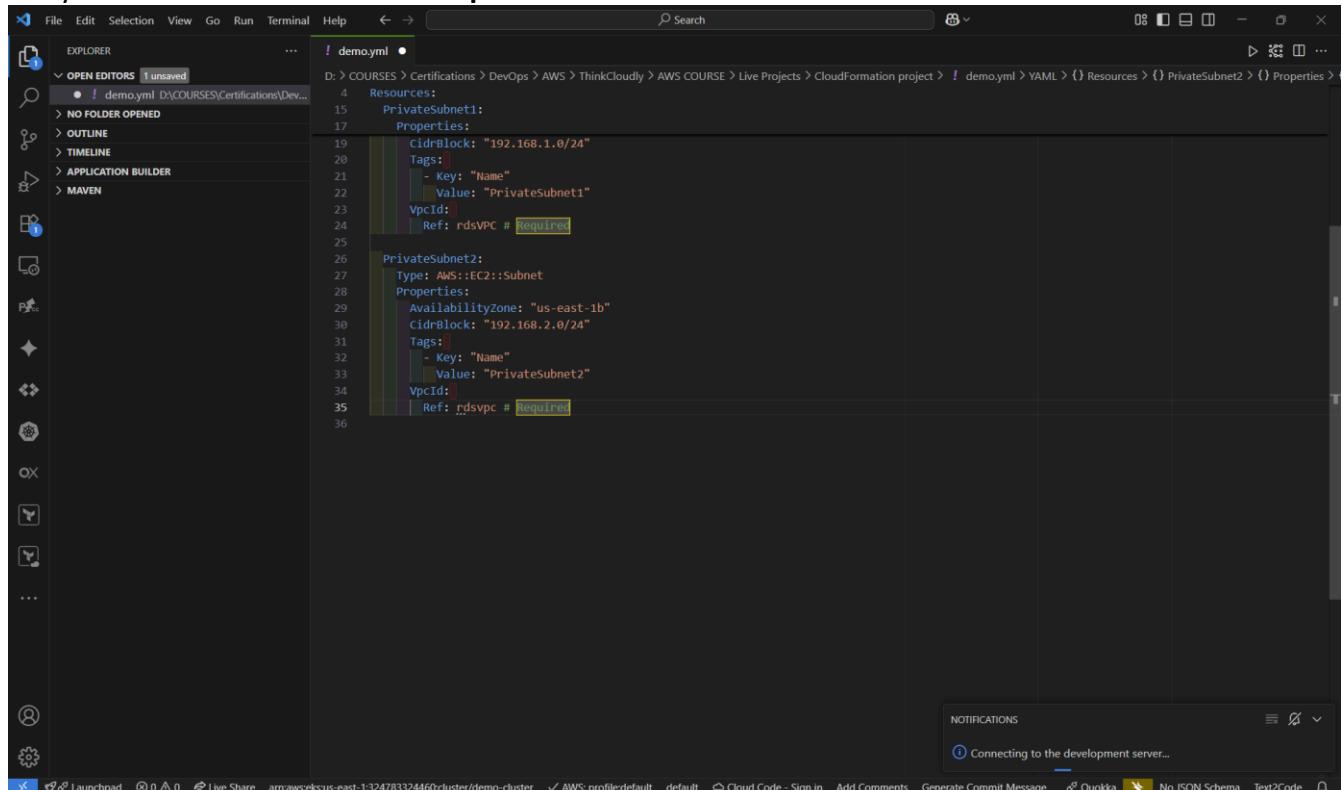
```

On “vpcId” enter “Ref: rdsVPC”



```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > PrivateSubnet2 > Properties >
  4 Resources:
  15   PrivateSubnet1:
  16     Properties:
  17       CidrBlock: "192.168.1.0/24"
  18       Tags:
  19         - Key: "Name"
  20           Value: "PrivateSubnet1"
  21       VpcId:
  22           Ref: rdsVPC # Required
  23
  24   PrivateSubnet2:
  25     Type: AWS::EC2::Subnet
  26     Properties:
  27       AvailabilityZone: "us-east-1b"
  28       CidrBlock: "192.168.2.0/24"
  29       Tags:
  30         - Key: "Name"
  31           Value: "PrivateSubnet2"
  32       VpcId:
  33           Ref: rdsVPC # Required ←
```

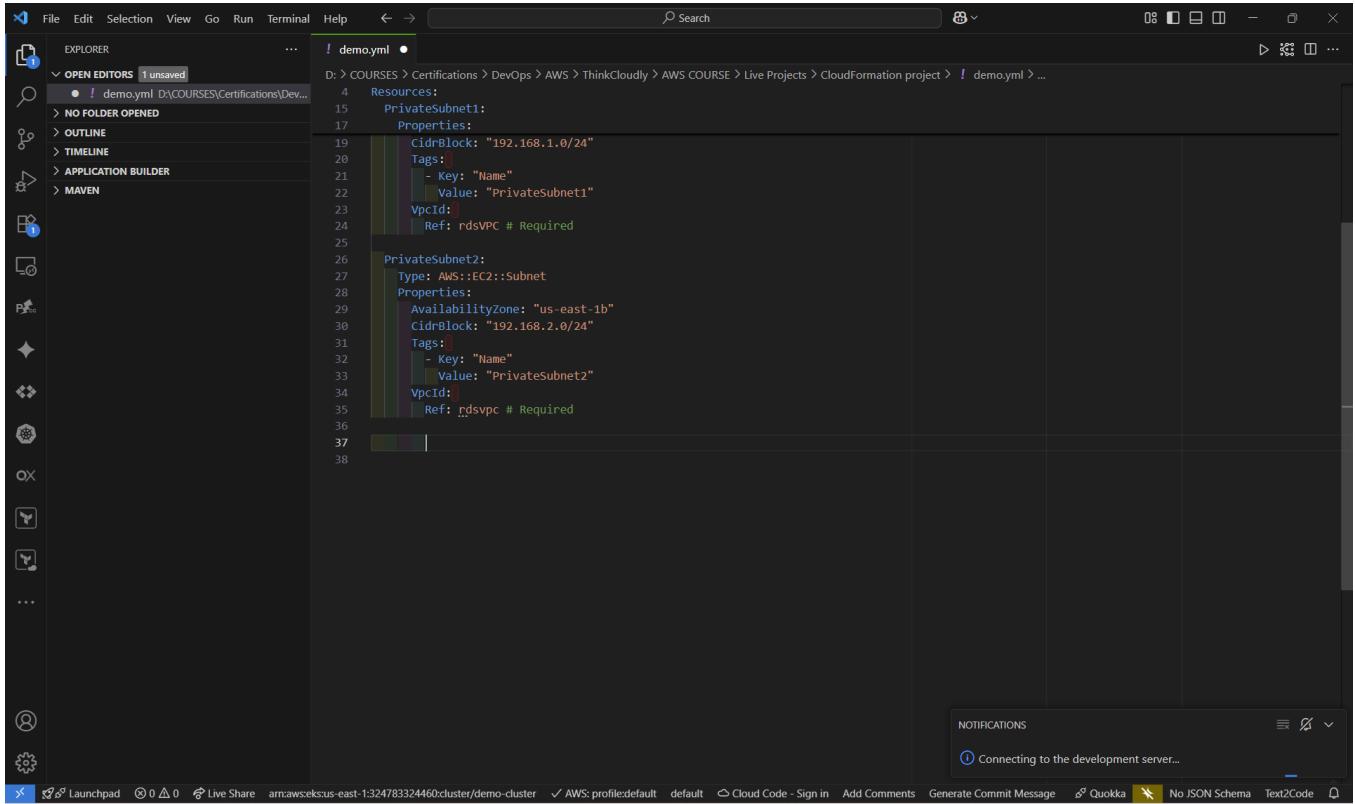
Put your cursor at the end of the “`vpcId`” line



```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > PrivateSubnet2 > Properties >
  4 Resources:
  15   PrivateSubnet1:
  16     Properties:
  17       CidrBlock: "192.168.1.0/24"
  18       Tags:
  19         - Key: "Name"
  20           Value: "PrivateSubnet1"
  21       VpcId:
  22           Ref: rdsVPC # required
  23
  24   PrivateSubnet2:
  25     Type: AWS::EC2::Subnet
  26     Properties:
  27       AvailabilityZone: "us-east-1b"
  28       CidrBlock: "192.168.2.0/24"
  29       Tags:
  30         - Key: "Name"
  31           Value: "PrivateSubnet2"
  32       VpcId:
  33           Ref: rdsVPC # required
```

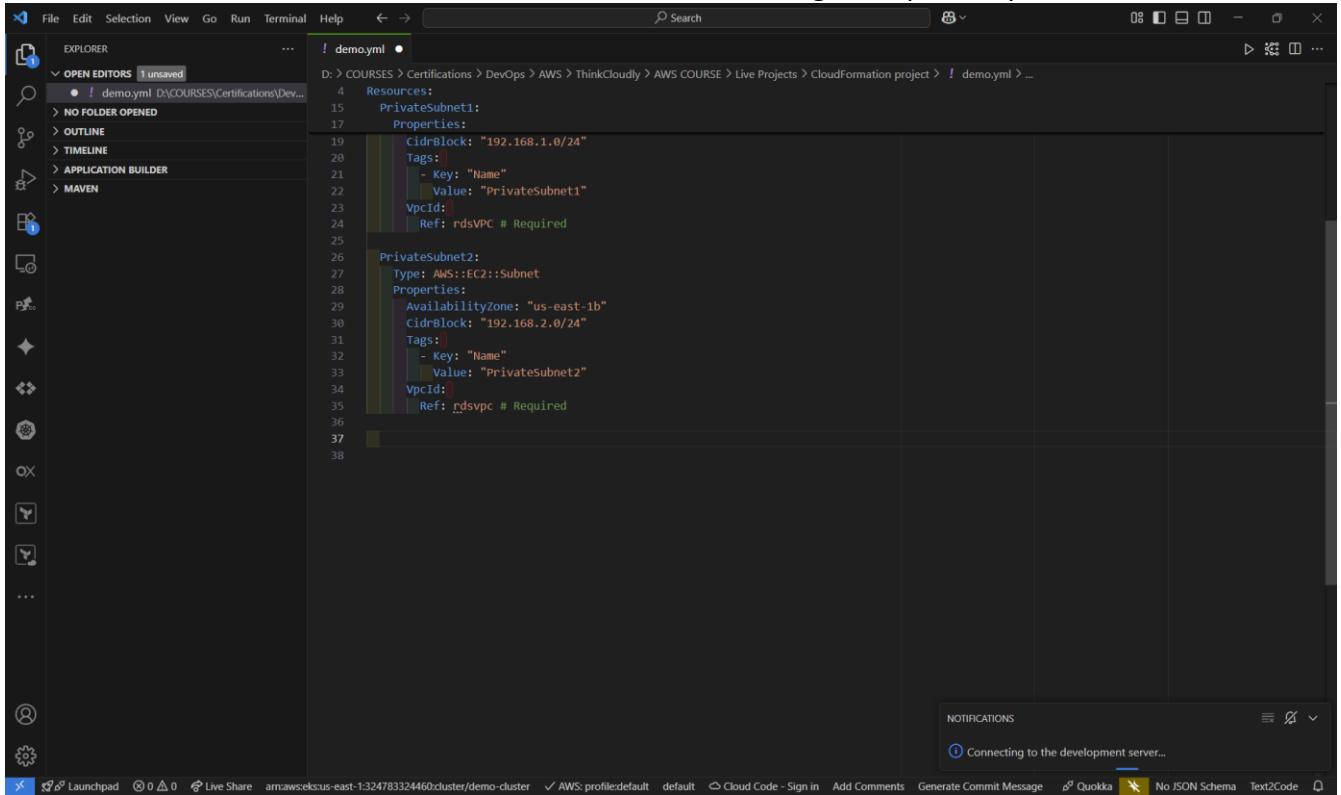
Press Enter twice

Prepared by Sidney Smith Ebott



```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yml ...
  4   Resources:
  15     PrivateSubnet1:
  16       Properties:
  17         CidrBlock: "192.168.1.0/24"
  18         Tags:
  19           - Key: "Name"
  20             Value: "PrivateSubnet1"
  21         VpcId: rdsVPC # Required
  22
  23     PrivateSubnet2:
  24       Type: AWS::EC2::Subnet
  25       Properties:
  26         AvailabilityZone: "us-east-1b"
  27         CidrBlock: "192.168.2.0/24"
  28         Tags:
  29           - Key: "Name"
  30             Value: "PrivateSubnet2"
  31         VpcId: rdsVPC # Required
  32
  33
  34
  35
  36
  37
  38
```

Move the cursor back to the line under “PrivateSubnet2” using backspace key



```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yml ...
  4   Resources:
  15     PrivateSubnet1:
  16       Properties:
  17         CidrBlock: "192.168.1.0/24"
  18         Tags: # Backspace here
  19           - Key: "Name"
  20             Value: "PrivateSubnet1"
  21         VpcId: rdsVPC # Required
  22
  23     PrivateSubnet2:
  24       Type: AWS::EC2::Subnet
  25       Properties:
  26         AvailabilityZone: "us-east-1b"
  27         CidrBlock: "192.168.2.0/24"
  28         Tags: # Backspace here
  29           - Key: "Name"
  30             Value: "PrivateSubnet2"
  31         VpcId: rdsVPC # Required
  32
  33
  34
  35
  36
  37
  38
```

We now have to create the “Security Group”, Type “**security**”

Prepared by Sidney Smith Ebott

The screenshot shows the AWS CloudFormation YAML editor interface. The left sidebar has icons for Explorer, Open Editors (1 unsaved), Outline, Timeline, Application Builder, and Maven. The main area displays a CloudFormation template with the following code:

```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > {} Resources
  4 Resources:
    5   PrivateSubnet1:
      6     Properties:
        7       CidrBlock: "192.168.1.0/24"
        8       Tags:
          9         - Key: "Name"
          10        Value: "PrivateSubnet1"
        11       VpcId:
        12         Ref: rdsVPC # Required
      13
    14   PrivateSubnet2:
      15     Type: AWS::EC2::Subnet
      16     Properties:
        17       AvailabilityZone: "us-east-1b"
        18       CidrBlock: "192.168.2.0/24"
        19       Tags:
          20         - Key: "Name"
          21         Value: "PrivateSubnet2"
        22       VpcId:
        23         Ref: rdsVPC # Required
      24
    25
  26
  27
  28
  29
  30
  31
  32
  33
  34
  35
  36
  37 security:
  38   - security-group-egress-cidr
   security-group-ingress-cidr
   securityhub-aggregatorv2
   securityhub-automationrule
   securityhub-automationrule_
   securityhub-configurationpol...
   securityhub-delegatedadmin
   securityhub-findingaggregat...
   securityhub-hub
   securityhub-hubv2
   securityhub-insight
   securityhub-organizationconfig...
```

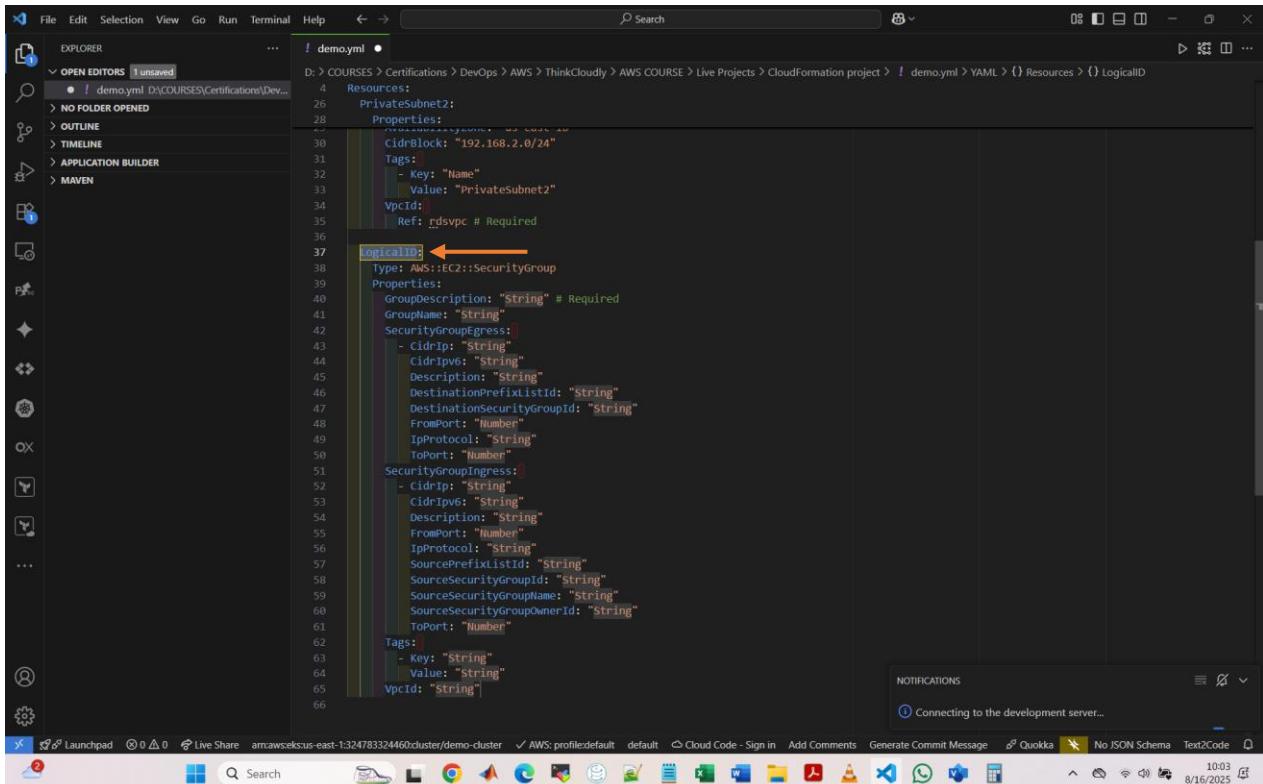
The 'security' section contains several AWS Service names: security-group-egress-cidr, security-group-ingress-cidr, securityhub-aggregatorv2, securityhub-automationrule, securityhub-automationrule_, securityhub-configurationpol..., securityhub-delegatedadmin, securityhub-findingaggregat..., securityhub-hub, securityhub-hubv2, securityhub-insight, and securityhub-organizationconfig....

Scroll down

The screenshot shows the AWS CloudFormation YAML editor interface, identical to the first one but with a red arrow pointing to the 'ec2-securitygroup' item in the 'security' section of the template.

```
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > {} Resources
  4 Resources:
    5   PrivateSubnet1:
      6     Properties:
        7       CidrBlock: "192.168.1.0/24"
        8       Tags:
          9         - Key: "Name"
          10        Value: "PrivateSubnet1"
        11       VpcId:
        12         Ref: rdsVPC # Required
      13
    14   PrivateSubnet2:
      15     Type: AWS::EC2::Subnet
      16     Properties:
        17       AvailabilityZone: "us-east-1b"
        18       CidrBlock: "192.168.2.0/24"
        19       Tags:
          20         - Key: "Name"
          21         Value: "PrivateSubnet2"
        22       VpcId:
        23         Ref: rdsVPC # Required
      24
    25
  26
  27
  28
  29
  30
  31
  32
  33
  34
  35
  36
  37 security:
  38   - securityhub-productsubscription
   securityhub-securitycontrol
   securityhub-standard
   securitylake-awslogsources
   securitylake-datalake
   securitylake-subscriber
   securitylake-subscribernotification
   ec2-securitygroup
   ec2-securitygroupingress
   ec2-securitygroupingress
   ec2-securitygroupvpcassociation
   emr-securityconfiguration...
```

Select "ec2-securitygroup"

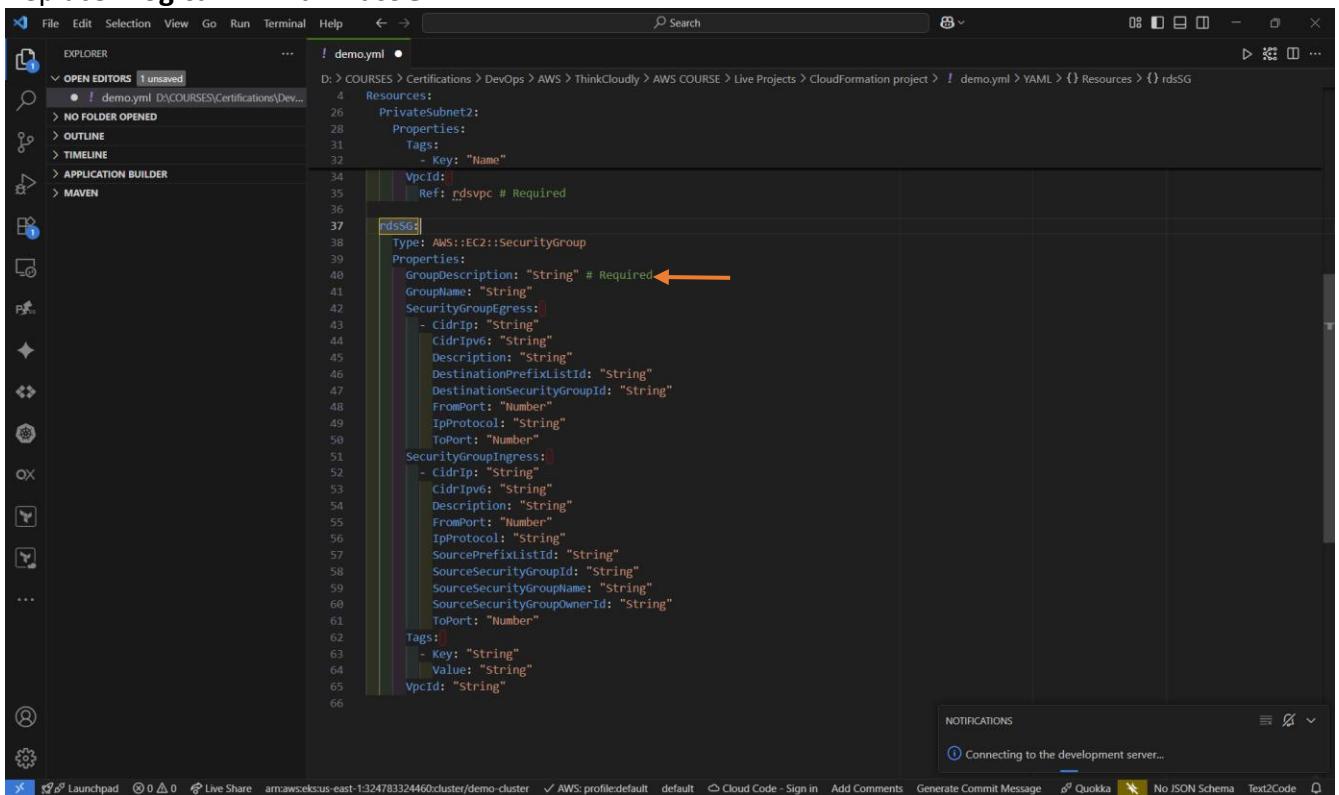


```

! demo.yml •
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yml > YAML > {} Resources > {} LogicalID
1 Resources:
2   PrivateSubnet2:
3     Properties:
4       CidrBlock: "192.168.2.0/24"
5       Tags:
6         - Key: "Name"
7           Value: "privateSubnet2"
8       VpcId:
9         Ref: rdsvpc # Required
10
11 LogicalID: ←
12   Type: AWS::EC2::SecurityGroup
13   Properties:
14     GroupDescription: "String" # Required
15     GroupName: "String"
16     SecurityGroupIngress:
17       - CidrIp: "String"
18         CidrIpv6: "String"
19         Description: "String"
20         DestinationPrefixListId: "String"
21         DestinationSecurityGroupId: "String"
22         FromPort: "Number"
23         IpProtocol: "String"
24         ToPort: "Number"
25     SecurityGroupEgress:
26       - CidrIp: "String"
27         CidrIpv6: "String"
28         Description: "String"
29         FromPort: "Number"
30         IpProtocol: "String"
31         SourcePrefixListId: "String"
32         SourceSecurityGroupName: "String"
33         SourceSecurityGroupOwnerId: "String"
34         ToPort: "Number"
35     Tags:
36       - Key: "String"
37         Value: "String"
38       VpcId: "String"
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66

```

Replace “LogicalID” with “rdsSG”



```

! demo.yml •
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yml > YAML > {} Resources > {} rdsSG
1 Resources:
2   PrivateSubnet2:
3     Properties:
4       Tags:
5         - Key: "Name"
6       VpcId:
7         Ref: rdsvpc # Required
8
9 rdsSG: ←
10   Type: AWS::EC2::SecurityGroup
11   Properties:
12     GroupDescription: "String" # Required ←
13     GroupName: "String"
14     SecurityGroupIngress:
15       - CidrIp: "String"
16         CidrIpv6: "String"
17         Description: "String"
18         DestinationPrefixListId: "String"
19         DestinationSecuritygroupId: "String"
20         FromPort: "Number"
21         IpProtocol: "String"
22         ToPort: "Number"
23     SecuritygroupEgress:
24       - CidrIp: "String"
25         CidrIpv6: "String"
26         Description: "String"
27         FromPort: "Number"
28         IpProtocol: "String"
29         SourcePrefixListId: "String"
30         SourceSecurityGroupOwnerId: "String"
31         SourceSecurityGroupName: "String"
32         ToPort: "Number"
33     Tags:
34       - Key: "String"
35         Value: "String"
36       VpcId: "String"
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66

```

On “GroupDescription” enter “rdsSG”

```

    Resources:
      PrivateSubnet2:
        Properties:
          Tags:
            - Key: "Name"
              Value: "PrivateSubnet2"
        VpcId: !Ref rdsVpc # Required

      rdssG:
        Type: AWS::EC2::SecurityGroup
        Properties:
          GroupDescription: "desc" # Required
          GroupName: "String"
          SecurityGroupIngress:
            - CidrIp: "String"
              CidrIpv6: "String"
              Description: "String"
              DestinationPrefixListId: "String"
              DestinationSecurityGroupId: "String"
              FromPort: "Number"
              IpProtocol: "String"
              ToPort: "Number"
          SecurityGroupIngress:
            - CidrIp: "String"
              CidrIpv6: "String"
              Description: "String"
              FromPort: "Number"
              IpProtocol: "String"
              SourcePrefixListId: "String"
              SourceSecurityGroupId: "String"
              SourceSecurityGroupName: "String"
              SourceSecurityGroupOwnerId: "String"
              ToPort: "Number"
          Tags:
            - Key: "String"
              Value: "String"
        VpcId: "String"
  
```

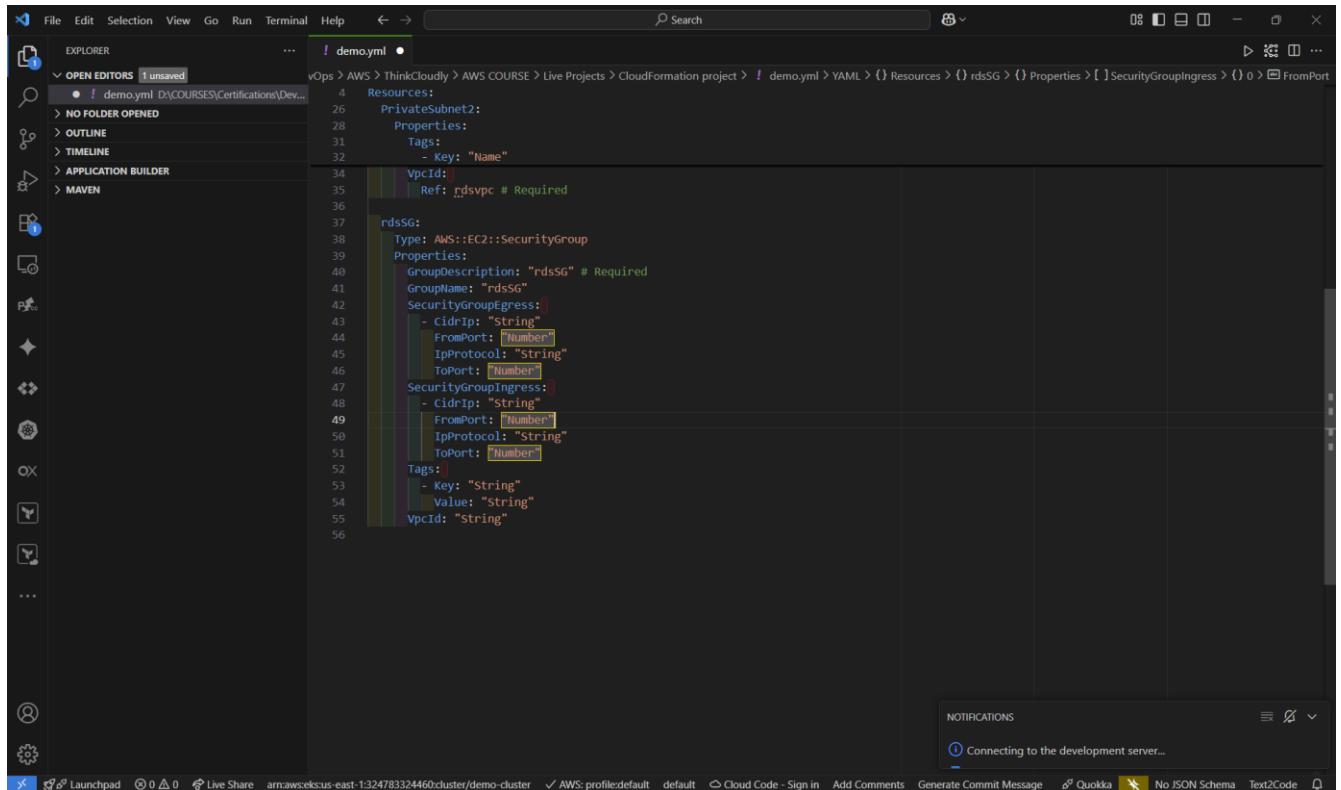
For “GroupName”, enter “rdsSG”

```

    Resources:
      PrivateSubnet2:
        Properties:
          Tags:
            - Key: "Name"
              Value: "PrivateSubnet2"
        VpcId: !Ref rdsVpc # Required

      rdssG:
        Type: AWS::EC2::SecurityGroup
        Properties:
          GroupDescription: "rdsSG" # Required
          GroupName: "rdsSG"
          SecurityGroupIngress:
            - CidrIp: "String"
              CidrIpv6: "String"
              Description: "String"
              DestinationPrefixListId: "String"
              DestinationSecurityGroupId: "String"
              FromPort: "Number"
              IpProtocol: "String"
              ToPort: "Number"
          SecurityGroupIngress:
            - CidrIp: "String"
              CidrIpv6: "String"
              Description: "String"
              FromPort: "Number"
              IpProtocol: "String"
              SourcePrefixListId: "String"
              SourceSecurityGroupId: "String"
              SourceSecurityGroupName: "String"
              SourceSecurityGroupOwnerId: "String"
              ToPort: "Number"
          Tags:
            - Key: "String"
              Value: "String"
        VpcId: "String"
  
```

Remove the lines that are not needed



The screenshot shows the AWS CloudFormation YAML editor in Visual Studio Code. The file being edited is `demo.yaml`. The code defines a security group (`rdssG`) with specific rules for ingress and egress traffic. The `SecurityGroupIngress` section contains two entries, each with a CIDR range of `0.0.0.0/0`, an IP protocol of `TCP`, and port ranges of `80` to `3306`. The `SecurityGroupEgress` section contains one entry with a CIDR range of `0.0.0.0/0`, an IP protocol of `TCP`, and port ranges of `80` to `3306`. The `Tags` section includes a tag with key `Name` and value `RDS`. The `VpcId` field is set to `rdsVpc`.

```
version: '2.0'
resources:
  rdsVpc:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
      EnableDnsSupport: true
      EnableDnsResolution: true
      InstanceTenancy: Isolated
      SubnetMappings:
        - SubnetType: Isolated
          VpcId: rdsVpc
          SubnetId: subnet-00000000
  rdssG:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: "rdssG" # Required
      GroupName: "rdssG"
      SecurityGroupEgress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "80"
          IpProtocol: "TCP"
          ToPort: "3306"
      SecurityGroupIngress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "80"
          IpProtocol: "TCP"
          ToPort: "3306"
      Tags:
        - Key: "Name"
          Value: "RDS"
      VpcId: rdsVpc # Required
```

Enter these details:

SG Egress CIDR: 0.0.0.0/0

SG Egress FromPort: 80

SG Egress ToPort: 80

SG Egress IPProtocol: TCP

SG Ingress CIDR: 0.0.0.0/0

SG Ingress FromPort: 3306

SG Ingress ToPort: 3306

SG Ingress IPProtocol: TCP

```

Resources:
  PrivateSubnet2:
    Properties:
      Tags:
        - Key: "Name"
          Value: "rdsSG"
      VpcId:
        Ref: rdsvp # Required

rdssg:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: "rdssg" # Required
    GroupName: "rdssg"
    SecuritygroupIngress:
      - CidrIp: "0.0.0.0/0"
        FromPort: "80"
        IpProtocol: "tcp"
        ToPort: "80"
      - CidrIp: "0.0.0.0/0"
        FromPort: "3306"
        IpProtocol: "tcp"
        ToPort: "3306"
    Tags:
      - Key: "String"
        Value: "String"
    VpcId: "String"
  
```

Under “Tag”, in “key” enter “Name” and in “value” enter “rdsSG”

```

Resources:
  PrivateSubnet2:
    Properties:
      Tags:
        - Key: "Name"
          Value: "rdsSG"
      VpcId:
        Ref: rdsvp # Required

rdssg:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: "rdssg" # Required
    GroupName: "rdssg"
    SecuritygroupIngress:
      - CidrIp: "0.0.0.0/0"
        FromPort: "80"
        IpProtocol: "tcp"
        ToPort: "80"
      - CidrIp: "0.0.0.0/0"
        FromPort: "3306"
        IpProtocol: "tcp"
        ToPort: "3306"
    Tags:
      - Key: "Name"
        Value: "rdsSG"
    VpcId: "String"
  
```

On “vpcId” enter “Ref: rdsvp”

```

File Edit Selection View Go Run Terminal Help < > Search
OPEN EDITORS 1 unsaved
demo.yaml
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > rdssg > Properties > VpcId
  Resources:
    PrivateSubnet2:
      Properties:
        Tags:
          - Key: "Name"
        VpcId:
          Ref: rdsvpc # Required
rdssg:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: "rdssg" # Required
    GroupName: "rdssg"
    SecuritygroupEgress:
      - CidrIp: "0.0.0.0/0"
        FromPort: "80"
        IpProtocol: "tcp"
        ToPort: "80"
    SecuritygroupIngress:
      - CidrIp: "0.0.0.0/0"
        FromPort: "3306"
        IpProtocol: "tcp"
        ToPort: "3306"
    Tags:
      - Key: "Name"
        Value: "rdssg"
    VpcId:
      Ref: rdsvpc # Required

```

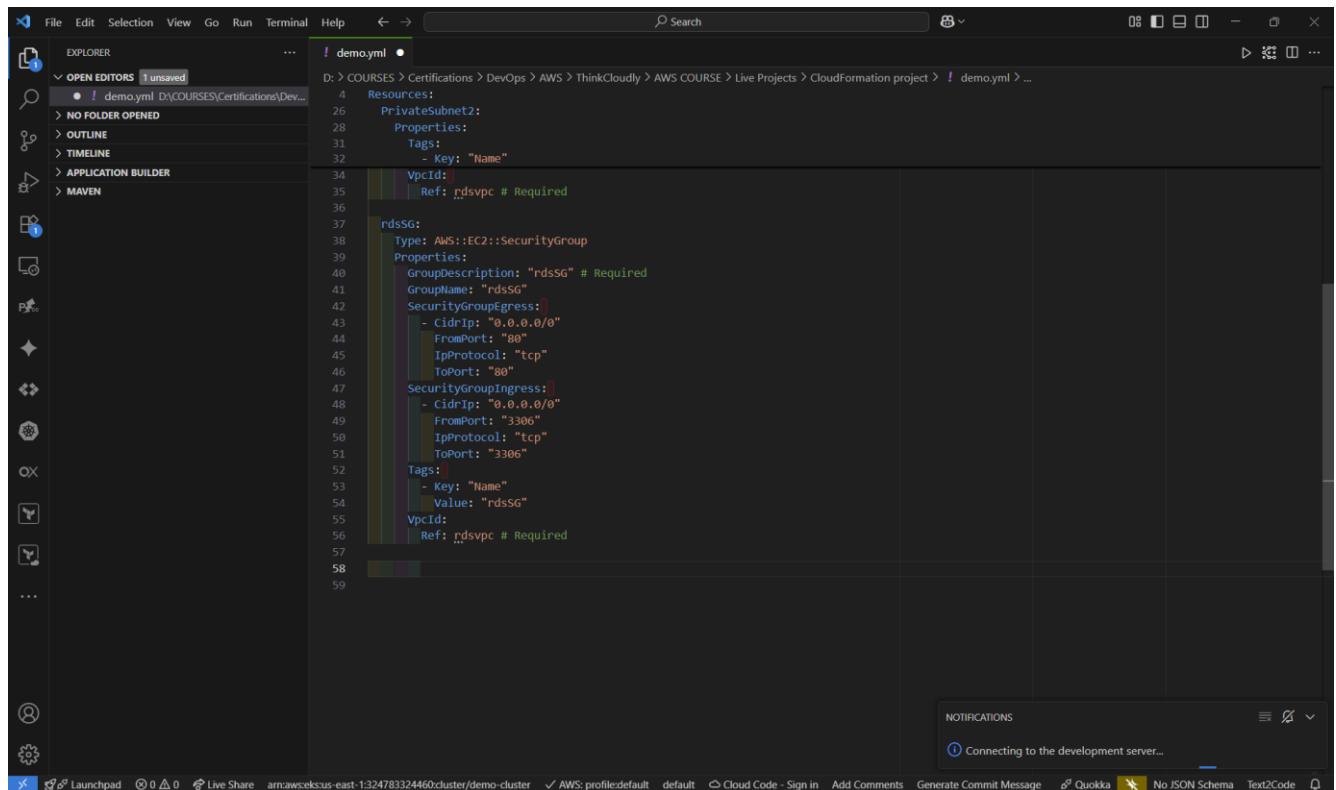
Put your cursor at the end of the “`vpcId`” line

```

File Edit Selection View Go Run Terminal Help < > Search
OPEN EDITORS 1 unsaved
demo.yaml
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > demo.yaml > YAML > Resources > rdssg > Properties > VpcId
  Resources:
    PrivateSubnet2:
      Properties:
        Tags:
          - Key: "Name"
        VpcId:
          Ref: rdsvpc # Required
rdssg:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: "rdssg" # Required
    GroupName: "rdssg"
    SecuritygroupEgress:
      - CidrIp: "0.0.0.0/0"
        FromPort: "80"
        IpProtocol: "tcp"
        ToPort: "80"
    SecuritygroupIngress:
      - CidrIp: "0.0.0.0/0"
        FromPort: "3306"
        IpProtocol: "tcp"
        ToPort: "3306"
    Tags:
      - Key: "Name"
        Value: "rdssg"
    VpcId:
      Ref: rdsvpc # Required

```

Press Enter twice

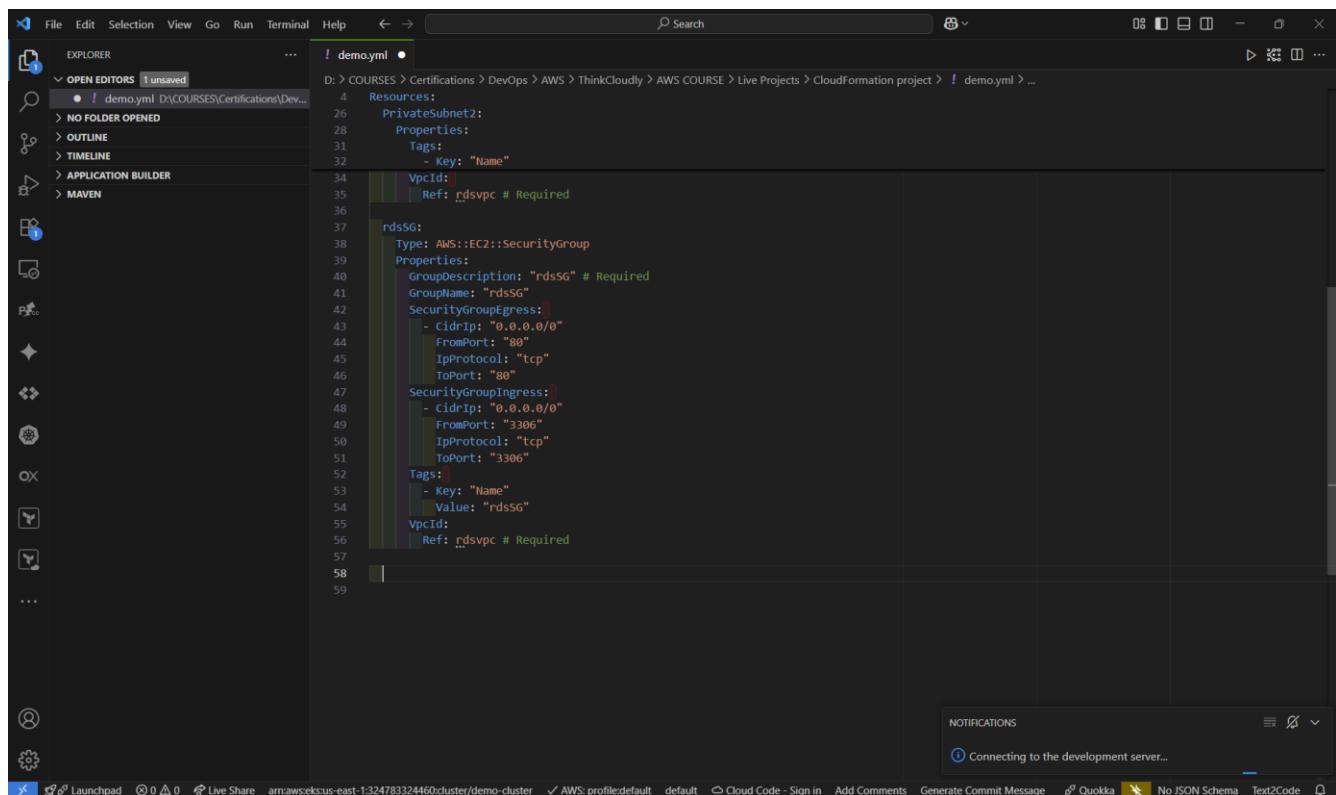


```

! demo.yaml
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yaml > ...
  Resources:
    PrivateSubnet2:
      Properties:
        Tags:
          - Key: "Name"
        VpcId:
          Ref: rdsvpc # Required
    rdssG:
      Type: AWS::EC2::SecurityGroup
      Properties:
        GroupDescription: "rdssG" # Required
        GroupName: "rdssG"
        SecurityGroupIngress:
          - CidrIp: "0.0.0.0/0"
            FromPort: "80"
            IpProtocol: "tcp"
            ToPort: "80"
        SecuritygroupIngress:
          - CidrIp: "0.0.0.0/0"
            FromPort: "3306"
            IpProtocol: "tcp"
            ToPort: "3306"
        Tags:
          - Key: "Name"
            Value: "rdssG"
        VpcId:
          Ref: rdsvpc # Required
  ...

```

Move the cursor back to the line under “**rdssG**” using backspace key

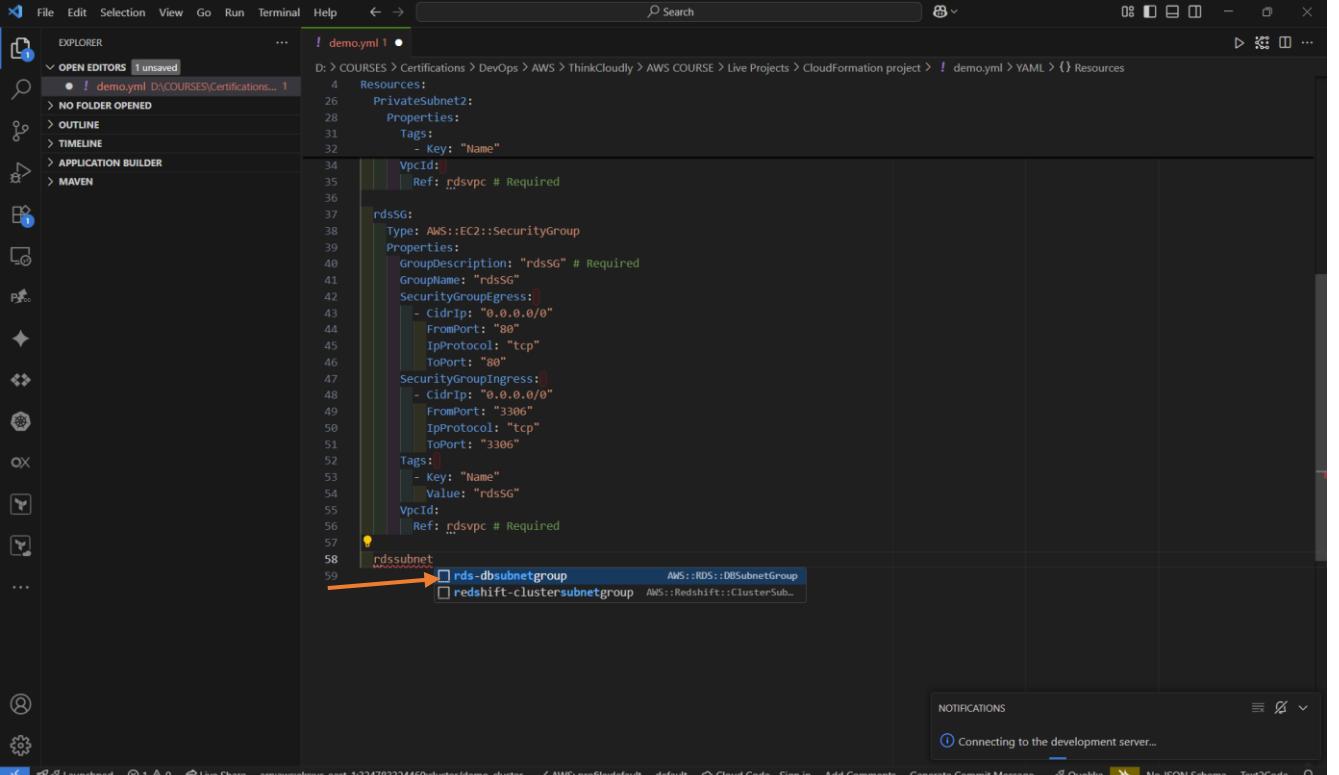


```

! demo.yaml
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yaml > ...
  Resources:
    PrivateSubnet2:
      Properties:
        Tags:
          - Key: "Name"
        VpcId:
          Ref: rdsvpc # Required
    rdssG:
      Type: AWS::EC2::SecurityGroup
      Properties:
        GroupDescription: "rdssG" # Required
        GroupName: "rdssG"
        SecurityGroupIngress:
          - CidrIp: "0.0.0.0/0"
            FromPort: "80"
            IpProtocol: "tcp"
            ToPort: "80"
        SecuritygroupIngress:
          - CidrIp: "0.0.0.0/0"
            FromPort: "3306"
            IpProtocol: "tcp"
            ToPort: "3306"
        Tags:
          - Key: "Name"
            Value: "rdssG"
        VpcId:
          Ref: rdsvpc # Required
  ...

```

Now, we have to create the subnet group. Type “**rdssubnet**”



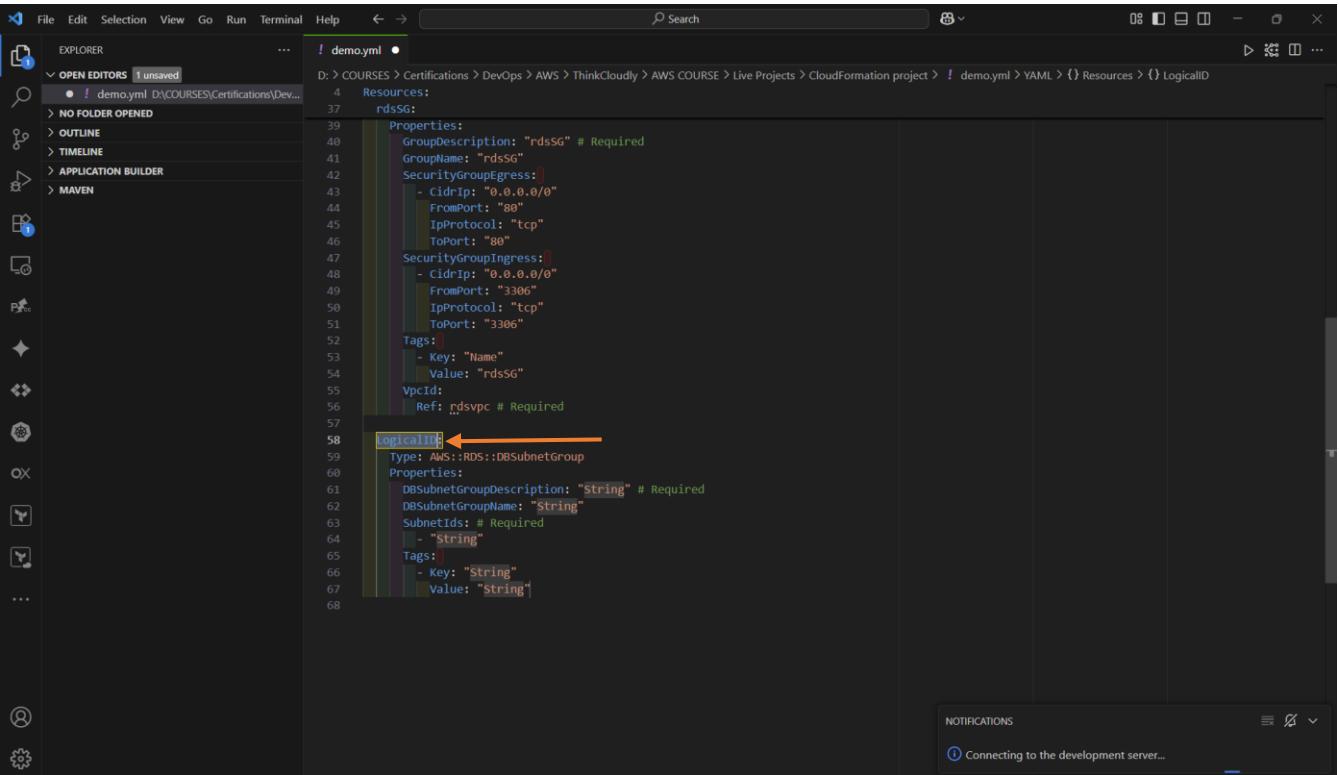
```

1 demo.yaml •
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yaml > YAML > {} Resources
4 Resources:
5   PrivateSubnet2:
6     Properties:
7       Tags:
8         - Key: "Name"
9       VpcId:
10      Ref: rdsvpc # Required
11
12 rdsSG:
13   Type: AWS::EC2::SecurityGroup
14   Properties:
15     GroupDescription: "rdsSG" # Required
16     GroupName: "rdsSG"
17     SecurityGroupEgress:
18       - CidrIp: "0.0.0.0/0"
19         FromPort: "80"
20         IpProtocol: "tcp"
21         ToPort: "80"
22     SecurityGroupIngress:
23       - CidrIp: "0.0.0.0/0"
24         FromPort: "3306"
25         IpProtocol: "tcp"
26         ToPort: "3306"
27     Tags:
28       - Key: "Name"
29         Value: "rdsSG"
30     VpcId:
31       Ref: rdsvpc # Required
32
33 rdssubnet:
34   Type: AWS::RDS::DBSubnetGroup
35   Properties:
36     DBSubnetGroupDescription: "rds-dbsubnetgroup"
37     DBSubnetGroupName: "rds-dbsubnetgroup"
38     SubnetIds: # Required
39       - "String"
40     Tags:
41       - Key: "String"
42         Value: "String"
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68

```

The screenshot shows the AWS CloudFormation YAML editor with the file 'demo.yaml' open. The code defines a security group 'rdsSG' with specific ingress and egress rules. It also defines a DB subnet group 'rdssubnet' with a single subnet ID. A red arrow points to the 'rdssubnet' section, highlighting the 'Type' field.

Select “rds-dbsubnetgroup”



```

1 demo.yaml •
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yaml > YAML > {} Resources > {} LogicalID
4 Resources:
5   rdsSG:
6     Properties:
7       GroupDescription: "rdsSG" # Required
8       GroupName: "rdsSG"
9       SecurityGroupEgress:
10      - CidrIp: "0.0.0.0/0"
11        FromPort: "80"
12        IpProtocol: "tcp"
13        ToPort: "80"
14     SecurityGroupIngress:
15       - CidrIp: "0.0.0.0/0"
16         FromPort: "3306"
17         IpProtocol: "tcp"
18         ToPort: "3306"
19     Tags:
20       - Key: "Name"
21         Value: "rdsSG"
22     VpcId:
23       Ref: rdsvpc # Required
24
25 logicalID:
26   Type: AWS::RDS::DBSubnetGroup
27   Properties:
28     DBSubnetGroupDescription: "rds-dbsubnetgroup"
29     DBSubnetGroupName: "rds-dbsubnetgroup"
30     SubnetIds: # Required
31       - "String"
32     Tags:
33       - Key: "String"
34         Value: "String"
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68

```

The screenshot shows the AWS CloudFormation YAML editor with the file 'demo.yaml' open. The code defines a security group 'rdsSG' and a DB subnet group 'logicalID'. A red arrow points to the 'logicalID' section, highlighting the 'Type' field.

Replace “LogicalID” with “rdsSubnetGroup”

The screenshot shows the VS Code interface with the 'demo.yaml' file open in the editor. The code defines an AWS CloudFormation stack with a single resource, 'rdsSG', which is a 'AWS::RDS::DBSubnetGroup'. The 'Properties' section includes 'GroupDescription' set to 'rdssG', 'GroupName' set to 'rdsSG', and 'SubnetIds' set to a list containing 'PrivateSubnet1' and 'PrivateSubnet2'. The 'SecurityGroupEgress' and 'SecurityGroupIngress' sections define port ranges for TCP traffic. The 'Tags' section adds a tag 'Name' with value 'rdssG'. The 'VpcId' field is set to a reference 'rdsvpc'. The 'rdssubnetGroup' section at the bottom is commented out.

```

    Resources:
        rdsSG:
            Properties:
                GroupDescription: "rdssG" # Required
                GroupName: "rdsSG"
                SecurityGroupEgress:
                    - CidrIp: "0.0.0.0/0"
                      FromPort: "80"
                      IpProtocol: "tcp"
                      ToPort: "80"
                SecurityGroupIngress:
                    - CidrIp: "0.0.0.0/0"
                      FromPort: "3306"
                      IpProtocol: "tcp"
                      ToPort: "3306"
                Tags:
                    - Key: "Name"
                      Value: "rdssG"
                VpcId:
                    Ref: rdsvpc # Required
            Type: AWS::RDS::DBSubnetGroup
            Properties:
                DBSubnetGroupDescription: "String" # Required
                DBSubnetGroupName: "String"
                SubnetIds: # Required
                    - "String"
                Tags:
                    - Key: "String"
                      Value: "String"
            rdssubnetGroup:
                Type: AWS::RDS::DBSubnetGroup
                Properties:
                    DBSubnetGroupDescription: "Group of Subnets" # Required
                    DBSubnetGroupName: "rdssubnetgroup"
                    SubnetIds: # Required
                        - "String"
                    Tags:
                        - Key: "String"
                          Value: "String"

```

For “**DBSubnetGroupDescription**” enter “**Group of Subnets**” and for “**DBSubnetGroupName**” enter “**rdsSubnetGroup**”

The screenshot shows the same VS Code interface with the 'demo.yaml' file open. The changes made are: 'Properties' now includes 'DBSubnetGroupDescription' set to 'Group of Subnets' and 'DBSubnetGroupName' set to 'rdsSubnetGroup'. The 'SubnetIds' field is also present in the 'Properties' section of the 'rdssubnetGroup' section, though it is commented out.

```

    Resources:
        rdsSG:
            Properties:
                GroupDescription: "rdssG" # Required
                GroupName: "rdsSG"
                SecurityGroupEgress:
                    - CidrIp: "0.0.0.0/0"
                      FromPort: "80"
                      IpProtocol: "tcp"
                      ToPort: "80"
                SecurityGroupIngress:
                    - CidrIp: "0.0.0.0/0"
                      FromPort: "3306"
                      IpProtocol: "tcp"
                      ToPort: "3306"
                Tags:
                    - Key: "Name"
                      Value: "rdssG"
                VpcId:
                    Ref: rdsvpc # Required
            Type: AWS::RDS::DBSubnetGroup
            Properties:
                DBSubnetGroupDescription: "Group of Subnets" # Required
                DBSubnetGroupName: "rdsSubnetGroup"
                SubnetIds: # Required
                    - "String"
                Tags:
                    - Key: "String"
                      Value: "String"
            rdssubnetGroup:
                Type: AWS::RDS::DBSubnetGroup
                Properties:
                    DBSubnetGroupDescription: "Group of Subnets" # Required
                    DBSubnetGroupName: "rdssubnetgroup"
                    SubnetIds: # Required
                        - "String"
                    Tags:
                        - Key: "String"
                          Value: "String"

```

For “**subnetIds**” enter these details:

- Ref: "PrivateSubnet1"
- Ref: "PrivateSubnet2"

```

Resources:
  rdssG:
    Properties:
      GroupDescription: "rdssG" # Required
      GroupName: "rdssG"
      SecurityGroupIngress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "80"
          IpProtocol: "tcp"
          ToPort: "80"
      SecurityGroupEgress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "3306"
          IpProtocol: "tcp"
          ToPort: "3306"
      Tags:
        - Key: "Name"
          Value: "rdsSG"
      VpcId:
        Ref: "rdsvc" # Required
  rdsSubnetGroup:
    Type: AWS::RDS::DBSubnetGroup
    Properties:
      DBSubnetGroupDescription: "Group of Subnets" # Required
      DBSubnetGroupName: "rdsSubnetGroup"
      SubnetIds: # Required
        - Ref: "PrivateSubnet1"
        - Ref: "PrivateSubnet2"
      Tags:
        - Key: "String"
          Value: "String"

```

Under “Tag”, in “key” enter “Name” and in “value” enter “**rdsSubnetGroup**”

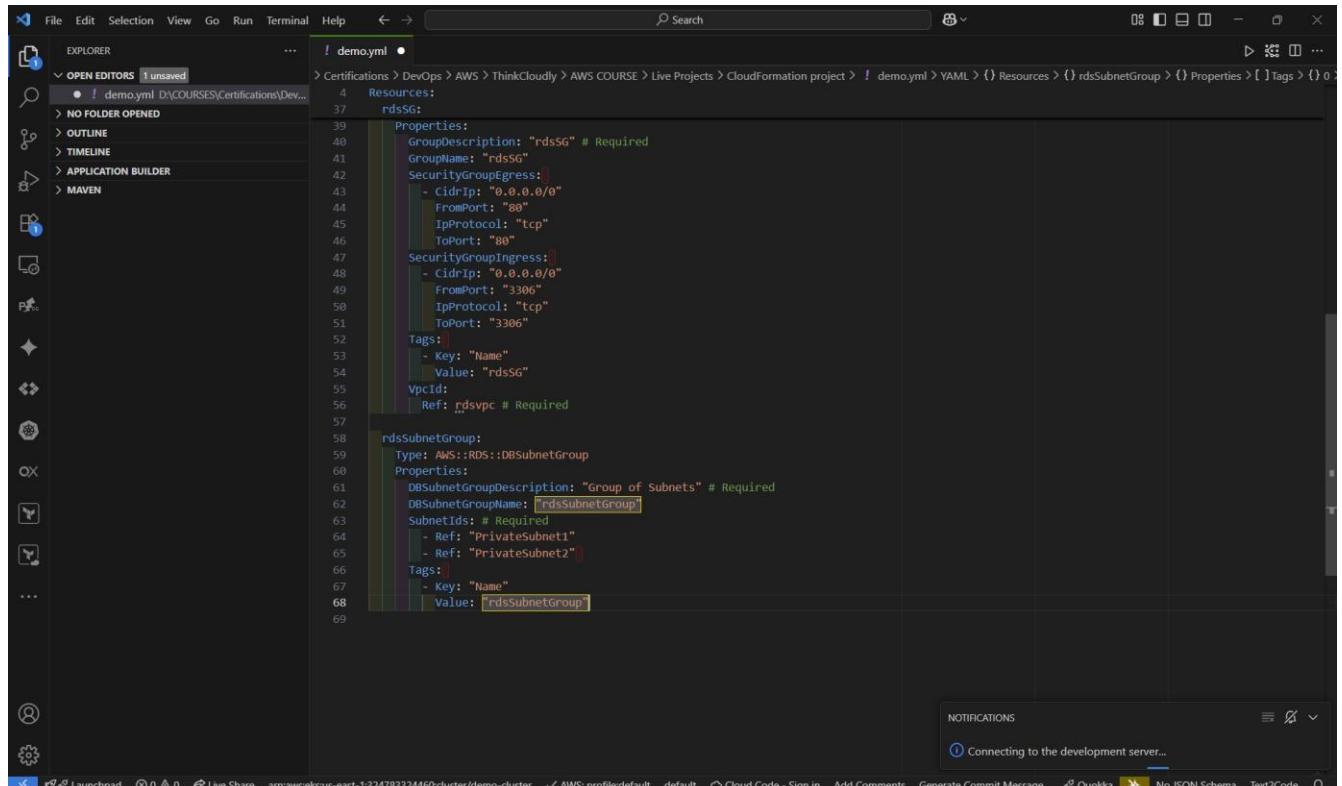
```

Resources:
  rdssG:
    Properties:
      GroupDescription: "rdssG" # Required
      GroupName: "rdssG"
      SecurityGroupIngress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "80"
          IpProtocol: "tcp"
          ToPort: "80"
      SecurityGroupEgress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "3306"
          IpProtocol: "tcp"
          ToPort: "3306"
      Tags:
        - Key: "Name"
          Value: "rdsSG"
        - Key: "rdsSubnetGroup"
          Value: "rdsSubnetGroup"
      VpcId:
        Ref: "rdsvc" # Required
  rdsSubnetGroup:
    Type: AWS::RDS::DBSubnetGroup
    Properties:
      DBSubnetGroupDescription: "Group of Subnets" # Required
      DBSubnetGroupName: "rdsSubnetGroup"
      SubnetIds: # Required
        - Ref: "PrivateSubnet1"
        - Ref: "PrivateSubnet2"
      Tags:
        - Key: "Name"
          Value: "String"

```

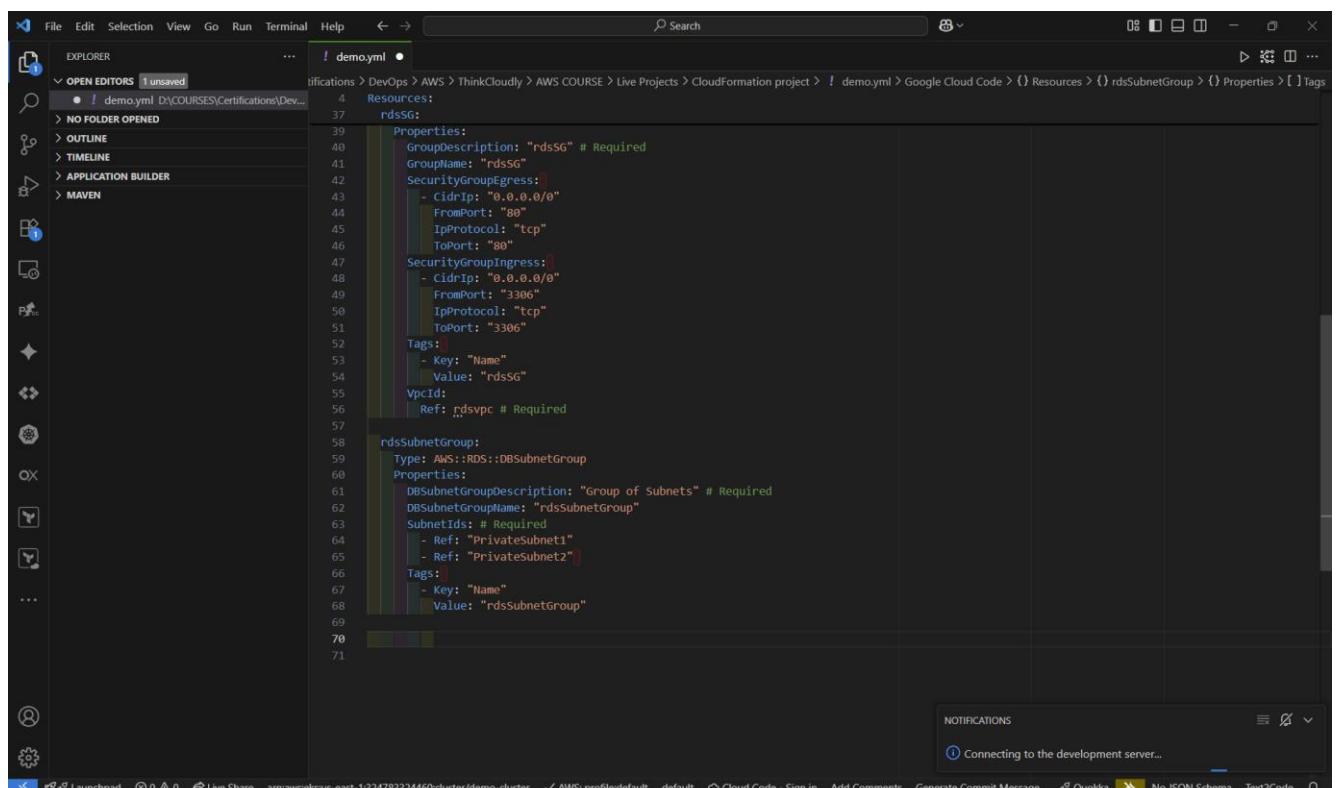
Then the next resource is our database instance. Put your cursor at the end of Value: “**rdsSubnetGroup**” line

Prepared by Sidney Smith Ebott



```
demo.yaml
Resources:
  rdsSG:
    Properties:
      GroupDescription: "rdssG" # Required
      GroupName: "rdssG"
      SecurityGroupIngress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "80"
          IpProtocol: "tcp"
          ToPort: "80"
      SecurityGroupIngress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "3306"
          IpProtocol: "tcp"
          ToPort: "3306"
      Tags:
        - Key: "Name"
          Value: "rdssG"
      VpcId:
        Ref: "rdsVPC" # Required
  rdsSubnetGroup:
    Type: AWS::RDS::DBSubnetGroup
    Properties:
      DBSubnetGroupDescription: "Group of Subnets" # Required
      DBSubnetGroupName: "rdsSubnetGroup"
      SubnetIds: # Required
        - Ref: "PrivateSubnet1"
        - Ref: "PrivateSubnet2"
      Tags:
        - Key: "Name"
          Value: "rdsSubnetGroup"
```

Press Enter twice



```
demo.yaml
Resources:
  rdsSG:
    Properties:
      GroupDescription: "rdssG" # Required
      GroupName: "rdssG"
      SecurityGroupIngress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "80"
          IpProtocol: "tcp"
          ToPort: "80"
      SecurityGroupIngress:
        - CidrIp: "0.0.0.0/0"
          FromPort: "3306"
          IpProtocol: "tcp"
          ToPort: "3306"
      Tags:
        - Key: "Name"
          Value: "rdssG"
      VpcId:
        Ref: "rdsVPC" # Required
  rdsSubnetGroup:
    Type: AWS::RDS::DBSubnetGroup
    Properties:
      DBSubnetGroupDescription: "Group of Subnets" # Required
      DBSubnetGroupName: "rdsSubnetGroup"
      SubnetIds: # Required
        - Ref: "PrivateSubnet1"
        - Ref: "PrivateSubnet2"
      Tags:
        - Key: "Name"
          Value: "rdsSubnetGroup"
```

Move the cursor back to the line under “**rdsSubnetGroup**” using backspace key

```

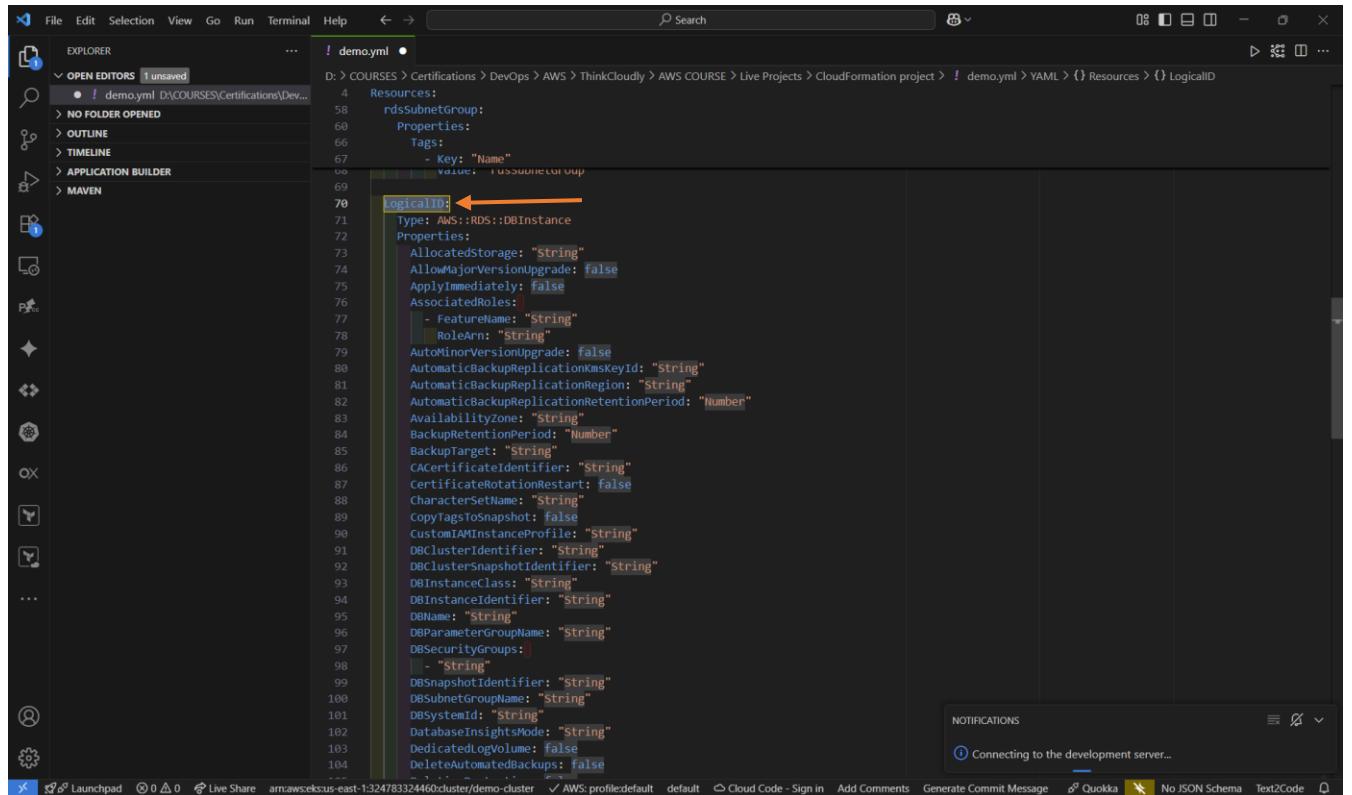
    rdsSG:
      Properties:
        GroupDescription: "rdssG" # Required
        GroupName: "rdssG"
        SecurityGroupEgress:
          - CidrIp: "0.0.0.0/0"
            FromPort: "80"
            IpProtocol: "tcp"
            ToPort: "80"
        SecurityGroupIngress:
          - CidrIp: "0.0.0.0/0"
            FromPort: "3306"
            IpProtocol: "tcp"
            ToPort: "3306"
        Tags:
          - Key: "Name"
            Value: "rdssG"
        VpcId:
          Ref: rdsvpc # Required
    rdsSubnetGroup:
      Type: AWS::RDS::DBSubnetGroup
      Properties:
        DBSubnetGroupDescription: "Group of Subnets" # Required
        DBSubnetGroupName: "rdsSubnetGroup"
        SubnetIds: # Required
          - Ref: "PrivateSubnet1"
          - Ref: "PrivateSubnet2"
        Tags:
          - Key: "Name"
            Value: "rdsSubnetGroup"
  
```

Type “rdsdbinstance”

```

    rdsSubnetGroup:
      Type: AWS::RDS::DBSubnetGroup
      Properties:
        DBSubnetGroupDescription: "Group of Subnets" # Required
        DBSubnetGroupName: "rdsSubnetGroup"
        SubnetIds: # Required
          - Ref: "PrivateSubnet1"
          - Ref: "PrivateSubnet2"
        Tags:
          - Key: "Name"
            Value: "rdsSubnetGroup"
    rdsdbinstance
  
```

Select “rds-dbinstance”

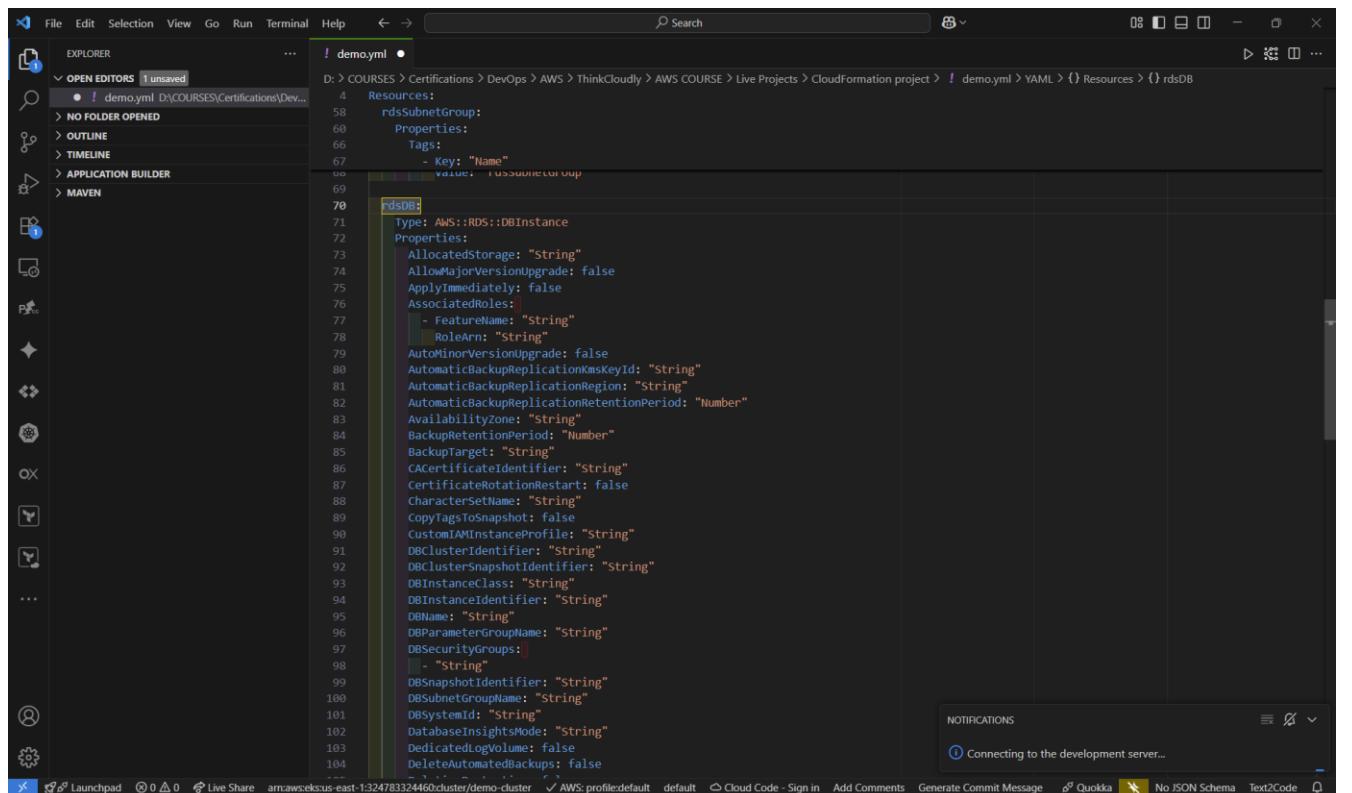


```

File Edit Selection View Go Run Terminal Help < > Search
OPEN EDITORS 1 unsaved
demo.yml •
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yml > YAML > {} Resources > {} LogicalID
4 Resources:
58   rdsSubnetGroup:
59     Properties:
60       Tags:
61         - Key: "Name"
62           Value: "rdssubnetgroup"
63
64 LogicalID: ← Red arrow here
65 Type: AWS::RDS::DBInstance
66 Properties:
67   AllocatedStorage: "String"
68   AllowMajorVersionUpgrade: false
69   ApplyImmediately: false
70   AssociatedRoles:
71     - FeatureName: "String"
72       RoleArn: "String"
73     AutoMinorVersionUpgrade: false
74     AutomaticBackupReplicationKmsKeyId: "String"
75     AutomaticBackupReplicationRegion: "String"
76     AutomaticBackupReplicationRetentionPeriod: "Number"
77     AvailabilityZone: "String"
78     BackupRetentionPeriod: "Number"
79     BackupTarget: "String"
80     CACertificateIdentifier: "String"
81     CertificateRotationRestart: false
82     CharacterSetName: "String"
83     CopyTagsToSnapshot: false
84     CustomIAMInstanceProfile: "String"
85     DBClusterIdentifier: "String"
86     DBClusterSnapshotIdentifier: "String"
87     DBInstanceClass: "String"
88     DBInstanceIdentifier: "String"
89     DBName: "String"
90     DBParameterGroupName: "String"
91     DBSecurityGroups:
92       - "String"
93     DBSnapshotIdentifier: "String"
94     DBSubnetGroupName: "String"
95     DBSystemId: "String"
96     DatabaseInsightsMode: "String"
97     DedicatedLogVolume: false
98     DeleteAutomatedBackups: false
99
100
101
102
103
104

```

Replace “LogicalID” with “rdsDB”

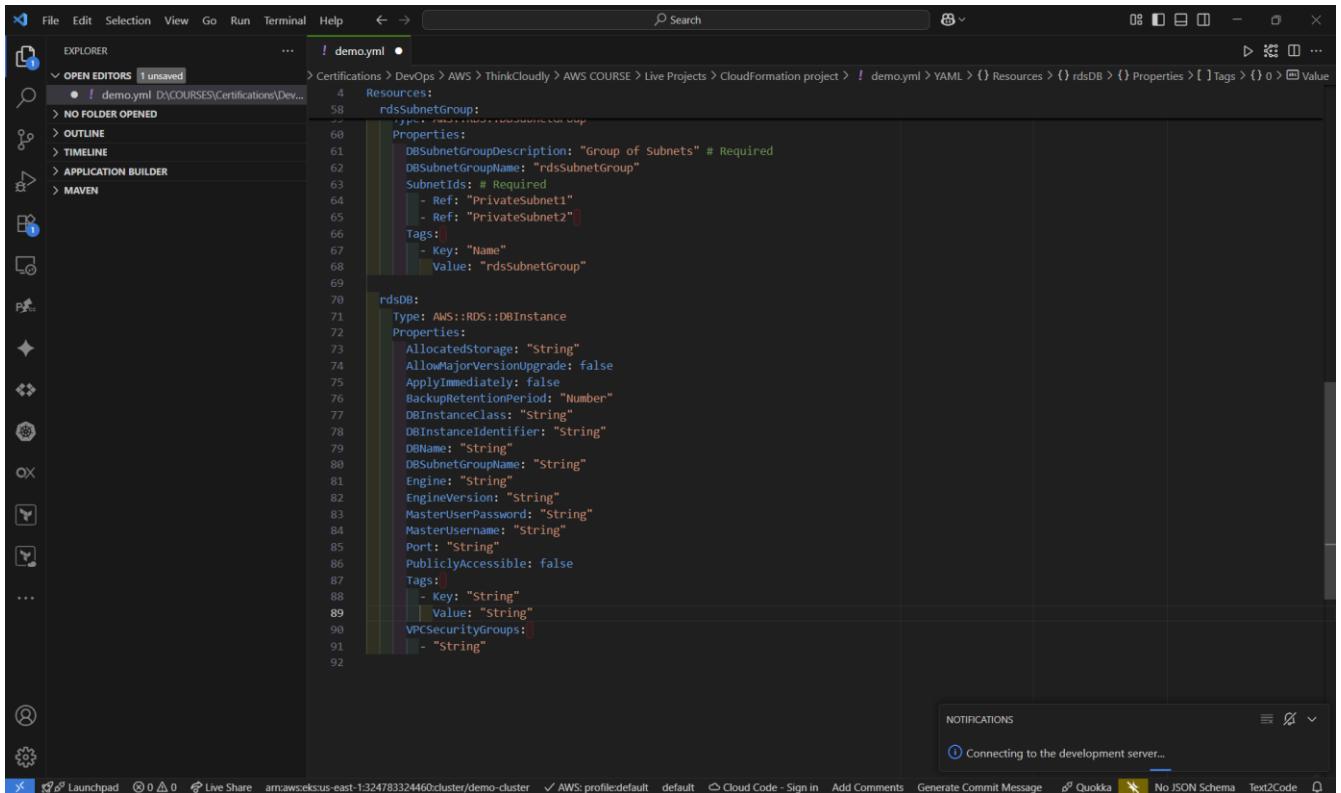


```

File Edit Selection View Go Run Terminal Help < > Search
OPEN EDITORS 1 unsaved
demo.yml •
D: > COURSES > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yml > YAML > {} Resources > {} rdsDB
4 Resources:
58   rdsSubnetGroup:
59     Properties:
60       Tags:
61         - Key: "Name"
62           Value: "rdssubnetgroup"
63
64 rdsDB: ← Red arrow here
65 Type: AWS::RDS::DBInstance
66 Properties:
67   AllocatedStorage: "String"
68   AllowMajorVersionUpgrade: false
69   ApplyImmediately: false
70   AssociatedRoles:
71     - FeatureName: "String"
72       RoleArn: "String"
73     AutoMinorVersionUpgrade: false
74     AutomaticBackupReplicationKmsKeyId: "String"
75     AutomaticBackupReplicationRegion: "String"
76     AutomaticBackupReplicationRetentionPeriod: "Number"
77     AvailabilityZone: "String"
78     BackupRetentionPeriod: "Number"
79     BackupTarget: "String"
80     CACertificateIdentifier: "String"
81     CertificateRotationRestart: false
82     CharacterSetName: "String"
83     CopyTagsToSnapshot: false
84     CustomIAMInstanceProfile: "String"
85     DBClusterIdentifier: "String"
86     DBClusterSnapshotIdentifier: "String"
87     DBInstanceClass: "String"
88     DBInstanceIdentifier: "String"
89     DBName: "String"
90     DBParameterGroupName: "String"
91     DBSecurityGroups:
92       - "String"
93     DBSnapshotIdentifier: "String"
94     DBSubnetGroupName: "String"
95     DBSystemId: "String"
96     DatabaseInsightsMode: "String"
97     DedicatedLogVolume: false
98     DeleteAutomatedBackups: false
99
100
101
102
103
104

```

Remove the line we do not need



The screenshot shows the AWS CloudFormation YAML editor interface. The left sidebar includes 'EXPLORER', 'OPEN EDITORS' (with one unsaved file), 'NO FOLDER OPENED', 'OUTLINE', 'TIMELINE', 'APPLICATION BUILDER', and 'MAVEN'. The main area displays a YAML template for creating an RDS instance and a subnet group:

```
Resources:
  rdsSubnetGroup:
    Type: AWS::RDS::DBSubnetGroup
    Properties:
      DBSubnetGroupDescription: "Group of Subnets" # Required
      DBSubnetGroupName: "rdsSubnetGroup"
      SubnetIds: # Required
        - Ref: "PrivateSubnet1"
        - Ref: "PrivateSubnet2"
      Tags:
        - Key: "Name"
          Value: "rdssubnetGroup"
  rdsDB:
    Type: AWS::RDS::DBInstance
    Properties:
      AllocatedStorage: "string"
      AllowMajorVersionUpgrade: false
      ApplyImmediately: false
      BackupRetentionPeriod: "Number"
      DBInstanceClass: "String"
      DBInstanceIdentifier: "String"
      DBName: "String"
      DBSubnetGroupName: "String"
      Engine: "String"
      EngineVersion: "String"
      MasterUserPassword: "String"
      MasterUsername: "String"
      Port: "String"
      PubliclyAccessible: false
      Tags:
        - Key: "String"
          Value: "String"
      VPCSecurityGroups:
        - "String"
```

The 'NOTIFICATIONS' panel at the bottom right shows a message: 'Connecting to the development server...'. The status bar at the bottom indicates the file path 'arn:aws:eks:sus-east-1:324783324460:cluster/demo-cluster' and the AWS profile 'default'.

Then enter these details:

Storage Space: 20 GB

Database Instance Class: Db.t2.micro

Database name: Amplededb

Database Engine: MySQL

Engine Version: 8.0.42

PubliclyAccessible: False

Username: awsuser

Password: Aws123456789

Port: 3306

AllowUpgradeVersion: False

BackUp Retention Period: 7 Days

```

! demo.yaml
D: > Certifications > DevOps > AWS > ThinkCloudly > AWS COURSE > Live Projects > CloudFormation project > ! demo.yaml > YAML > {} Resources > {} Properties > [ ] Tags
4   Resources:
58     rdsSubnetGroup:
59       Type: AWS::RDS::DBSubnetGroup
60       Properties:
61         DBSubnetGroupDescription: "Group of Subnets" # Required
62         DBSubnetGroupName: "rdssubnetGroup"
63         SubnetIds: # Required
64           - Ref: "PrivateSubnet1"
65           - Ref: "PrivateSubnet2"
66         Tags:
67           - Key: "Name"
68           - Value: "rdssubnetgroup"
69
70     rdsDB:
71       Type: AWS::RDS::DBInstance
72       Properties:
73         AllocatedStorage: "20"
74         AllowMajorVersionUpgrade: false
75         ApplyImmediately: false
76         BackupRetentionPeriod: "7"
77         DBInstanceClass: "db.t3.micro"
78         DBInstanceIdentifier: "myRDSInstance"
79         DBName: "AmpledB"
80         DBSubnetGroupName:
81           Ref: rdsSubnetGroup # Required
82         Engine: "MySQL"
83         EngineVersion: "8.0.42"
84         MasterUserPassword: "Aws123456789"
85         MasterUsername: "awsuser"
86         Port: "3306"
87         PubliclyAccessible: false
88         Tags:
89           - Key: "Name"
90           - Value: "rdsDB"
91         VPCSecurityGroups:
92           - Ref: rdssG
93

```

Save the file by clicking on File -> Save

```

AWSTemplateFormatVersion: 2010-09-09
Description: Live project cloud formation

```

Resources:

rdsVPC:

Type: AWS::EC2::VPC

Properties:

CidrBlock: "192.168.0.0/16"

EnableDnsHostnames: false

EnableDnsSupport: false

Tags:

- Key: "Name"

Value: "rds_VPC"

PrivateSubnet1:

Type: AWS::EC2::Subnet

Properties:

AvailabilityZone: "us-east-1a"

CidrBlock: "192.168.1.0/24"

Tags:

- Key: "Name"

Value: "PrivateSubnet1"

VpcId:

Ref: rdsVPC #The Ref value (rdsVPC) should match with the name of the VPC resource (rdsVPC)

PrivateSubnet2:

```
Type: AWS::EC2::Subnet
Properties:
  AvailabilityZone: "us-east-1b"
  CidrBlock: "192.168.2.0/24"
  Tags:
    - Key: "Name"
      Value: "PrivateSubnet2"
VpcId:
  Ref: rdsVPC # Required
```

rdsSG:←

```
Type: AWS::EC2::SecurityGroup
Properties:
  GroupDescription: "rdsSG" # Required
 GroupName: "rdsSG"
  SecurityGroupEgress:
    - CidrIp: "0.0.0.0/0"
      FromPort: "80"
      IpProtocol: "tcp"
      ToPort: "80"
  SecurityGroupIngress:
    - CidrIp: "0.0.0.0/0"
      FromPort: "3306"
      IpProtocol: "tcp"
      ToPort: "3306"
  Tags:
    - Key: "Name"
      Value: "rdsSG"
VpcId:
  Ref: rdsVPC # Required
```

rdsSubnetGroup:

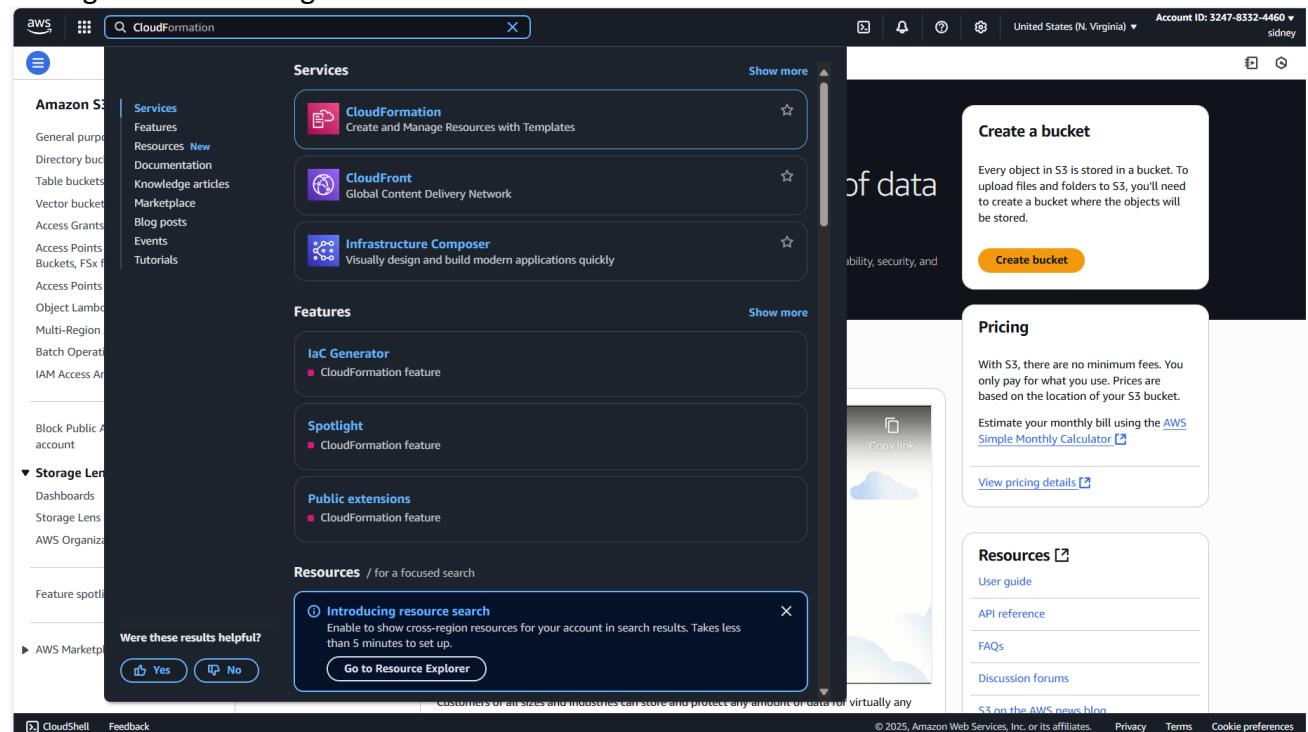
```
Type: AWS::RDS::DBSubnetGroup
Properties:
  DBSubnetGroupDescription: "Group of Subnets" # Required
  DBSubnetGroupName: "rdsSubnetGroup"
  SubnetIds: # Required
    - Ref: "PrivateSubnet1"
    - Ref: "PrivateSubnet2"
  Tags:
    - Key: "Name"
      Value: "rdsSubnetGroup"
```

rdsDB:

```
Type: AWS::RDS::DBInstance
Properties:
  AllocatedStorage: "20"
  AllowMajorVersionUpgrade: false
  ApplyImmediately: false
  BackupRetentionPeriod: "7"
```

```
DBInstanceClass: "db.t3.micro"
DBInstanceIdentifier: "myRDSInstance"
DBName: "Ampledbs"
DBSubnetGroupName:
  Ref: rdsSubnetGroup  # Required
Engine: "MySQL"
EngineVersion: "8.0.42"
MasterUserPassword: "Aws123456789"
MasterUsername: "awsuser"
Port: "3306"
PubliclyAccessible: false
Tags:
  - Key: "Name"
    Value: "rdsDB"
VPCSecurityGroups:
  - Ref: rdsSG
```

Then go to AWS Management console and search for “CloudFormation”



Click on “CloudFormation” under services

Prepared by Sidney Smith Ebot

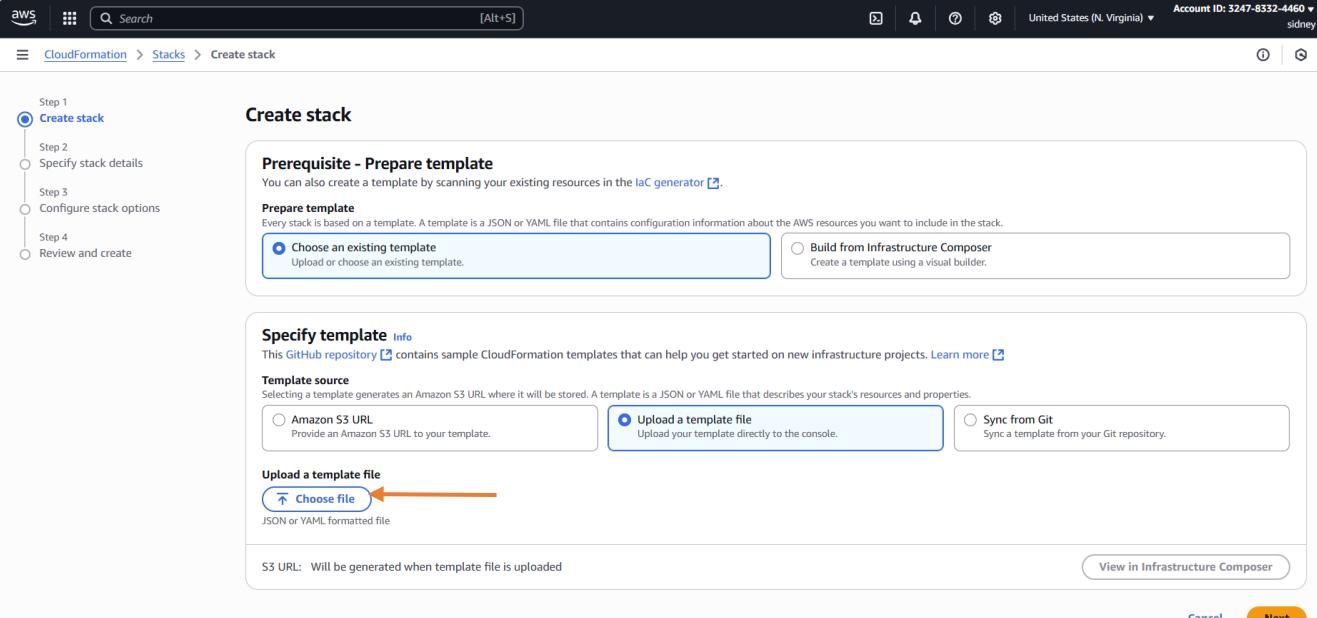
The screenshot shows the AWS CloudFormation Stacks page. On the left, there's a sidebar with sections like 'CloudFormation', 'Stacks', 'Infrastructure Composer', 'Hooks', 'Registry', 'Feedback', and 'Spotlight'. The main area is titled 'Stacks (0)' and shows a table with columns 'Stack name', 'Status', and 'Created time'. A message 'No stacks' and 'No stacks to display' is centered. At the bottom right of the table area is a large orange 'Create stack' button, which has a red arrow pointing to it from the left.

Click on “Create Stack”

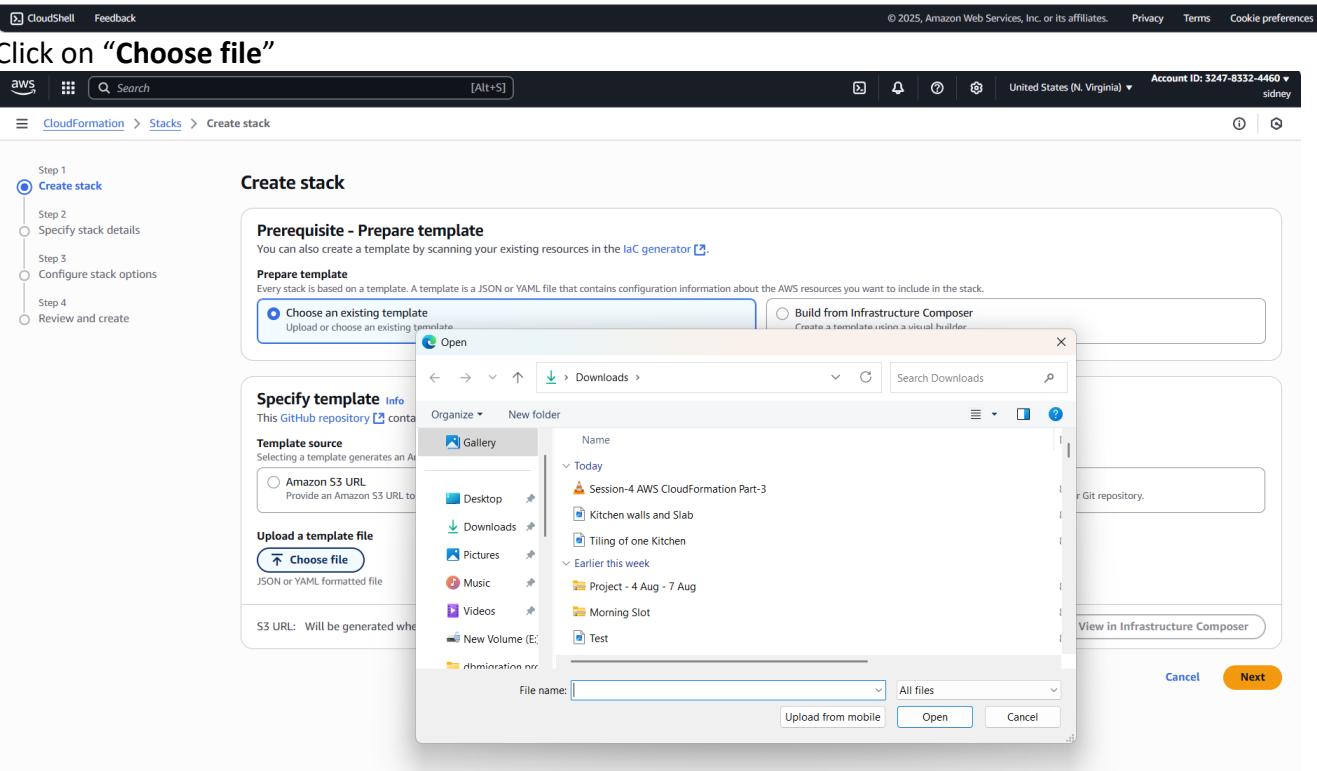
The screenshot shows the 'Create stack' wizard, Step 1: Prerequisites - Prepare template. On the left, a sidebar lists steps: Step 1 (Create stack, selected), Step 2, Step 3, Step 4, and Review and create. The main area has two options: 'Choose an existing template' (selected) and 'Build from Infrastructure Composer'. Below is a section for 'Specify template' with three options: 'Amazon S3 URL' (selected), 'Upload a template file' (highlighted with an orange arrow), and 'Sync from Git'. The 'Amazon S3 URL' field contains 'https://'. At the bottom right are 'Cancel' and 'Next' buttons.

On “Template Source”, select “Upload a template file”

Prepared by Sidney Smith Ebot



The screenshot shows the AWS CloudFormation 'Create stack' wizard at Step 1: Prerequisite - Prepare template. The 'Choose an existing template' option is selected, with a tooltip 'Upload or choose an existing template.' A blue arrow points to the 'Choose file' button in the 'Upload a template file' section.



The screenshot shows the AWS CloudFormation 'Create stack' wizard at Step 1: Prerequisite - Prepare template. The 'Choose an existing template' option is selected. A file selection dialog is open over the 'Choose file' button, showing a list of files in the Downloads folder. The dialog includes a search bar, file list, and navigation buttons.

Navigate to where our configuration file is stored

Prepared by Sidney Smith Ebot

The screenshot shows the AWS CloudFormation 'Create stack' wizard at Step 1: Prerequisite - Prepare template. On the left, a sidebar lists steps: Step 1 (Create stack), Step 2 (Specify stack details), Step 3 (Configure stack options), Step 4 (Review and create). The main area shows the 'Prerequisite - Prepare template' step. It includes a note about creating a template by scanning existing resources or using the IaC generator. Two options are available: 'Choose an existing template' (selected) and 'Build from Infrastructure Composer'. A file dialog is overlaid on the page, showing a file selection interface. The file 'demo' is selected in the list, which is described as a YML File. Other files listed include 'new' (YML File) and 'Problem (1)' (Microsoft Word Document). The file dialog has a search bar, a preview pane, and buttons for 'Open' and 'Cancel'.

The screenshot shows the AWS CloudFormation 'Create stack' wizard at Step 1: Prerequisite - Prepare template. The sidebar shows steps: Step 1 (Create stack), Step 2 (Specify stack details), Step 3 (Configure stack options), Step 4 (Review and create). The main area shows the 'Prerequisite - Prepare template' step. It includes a note about creating a template by scanning existing resources or using the IaC generator. Two options are available: 'Choose an existing template' (selected) and 'Build from Infrastructure Composer'. Below this, there is a 'Template source' section with an 'Amazon S3 URL' option and an 'Upload a template file' section. In the 'Upload a template file' section, a file input field contains 'demo.yml', which is described as a JSON or YAML formatted file. There are also options for 'Sync from Git' and 'View in Infrastructure Composer'. At the bottom, there are 'Cancel' and 'Next' buttons.

Click on “Next”

Prepared by Sidney Smith Ebot

The screenshot shows the 'Specify stack details' step of the CloudFormation wizard. The left sidebar shows steps 1 through 4. Step 2, 'Specify stack details', is selected and highlighted in blue. The main area has a title 'Provide a stack name' and a 'Stack name' input field containing 'Enter a stack name'. Below it is a note: '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: 0/128.' To the right of the input field is a note: 'Parameters' with a sub-note: 'Parameters are defined in your template and allow you to input custom values when you create or update a stack.' Below that is a section titled 'No parameters' with the note: 'There are no parameters defined in your template.' At the bottom right are buttons for 'Cancel', 'Previous', and 'Next'.

Give the stack a name, I will call it “demostack”

The screenshot shows the 'Specify stack details' step of the CloudFormation wizard. The left sidebar shows steps 1 through 4. Step 2, 'Specify stack details', is selected and highlighted in blue. The main area has a title 'Provide a stack name' and a 'Stack name' input field containing 'demostack'. Below it is a note: '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: 9/128.' To the right of the input field is a note: 'Parameters' with a sub-note: 'Parameters are defined in your template and allow you to input custom values when you create or update a stack.' Below that is a section titled 'No parameters' with the note: 'There are no parameters defined in your template.' At the bottom right are buttons for 'Cancel', 'Previous', and 'Next'.

Click on “Next”

The screenshot shows the 'Configure stack options' step of the CloudFormation wizard. The left sidebar shows steps 1 through 4. Step 3, 'Configure stack options', is selected and highlighted in blue. The main area has a title 'Configure stack options'. It contains three sections: 'Tags - optional' (with a note about adding up to 50 unique tags), 'Permissions - optional' (with a note about choosing an IAM role for CloudFormation operations), and 'Stack failure options' (with a note about behavior on provisioning failure and two radio button options: 'Roll back all stack resources' and 'Preserve successfully provisioned resources'). At the bottom right are buttons for 'Cancel', 'Previous', and 'Next'.

Prepared by Sidney Smith Ebot

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.

Additional settings
You can set additional options for your stack, like notification options and a stack policy. Learn more [\[?\]](#)

▶ Stack policy - optional
Defines the resources that you want to protect from unintentional updates during a stack update.

▶ Rollback configuration - optional
Specify alarms for CloudFormation to monitor when creating and updating the stack. If the operation breaches an alarm threshold, CloudFormation rolls it back.

▶ Notification options - optional
Specify a new or existing Amazon Simple Notification Service topic where notifications about stack events are sent.

▶ Stack creation options - optional
Specify the timeout and termination protection options for stack creation.

Cancel Previous Next

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

Click on “Next”

Review and create

Step 1: Specify template

Prerequisite - Prepare template

Template
Template is ready

Template

Template URL
<https://s3.us-east-1.amazonaws.com/cf-templates-10oyihrn9dl3s-us-east-1/2025-08-16T040311.516Z/48-demo.yml>

Stack description
awsproject2

Step 2: Specify stack details

Provide a stack name

Stack name
demostack

Parameters

Search Key ▲ | Value ▼

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

Review

Prepared by Sidney Smith Ebot

The screenshot shows the 'Create stack' wizard in the AWS CloudFormation console. The 'Monitoring time' field is set to '-'. The 'CloudWatch alarm ARN' field is also set to '-'. Under 'Notification options', there is a 'SNS topic ARN' field which is empty, resulting in a message: 'No notification options' and 'There are no notification options defined'. In the 'Stack creation options' section, 'Timeout' is set to '-' and 'Termination protection' is set to 'Deactivated'. A 'Quick-create link' section provides a URL for sharing the stack configuration. At the bottom right are 'Cancel', 'Previous', and 'Submit' buttons.

Click on "Submit"

The screenshot shows the 'Resources' tab for the 'demostack' stack. On the left, the 'Stacks' list shows one stack named 'demostack' with a status of 'CREATE_IN_PROGRESS'. The main area displays the 'Resources' table with one item:

Logical ID	Physical ID	Type	Status
rdsVPC	vpc-0babd957d940f047a	AWS::EC2::VPC	CREATE_IN_PROGRESS

The stack is being created

Prepared by Sidney Smith Ebot

The screenshot shows the AWS CloudFormation console with the URL <https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/resources?stackId=arn%3Aaws%3Acloudformation%3Aus-east-1%3A324783324460%3Astac...>. The page displays a stack named 'demostack' created on 2025-08-16 at 00:53:35 UTC-0400 with a status of 'CREATE_COMPLETE'. The 'Resources' tab is selected, showing a list of six resources: PrivateSubnet1, PrivateSubnet2, rdsDB, rdsSG, rdsSubnetGroup, and rdsVPC. Each resource has its logical ID, physical ID, type, status (all shown as 'CREATE_COMPLETE'), and module.

The stack has been created. Click on “Events” tab

The screenshot shows the AWS CloudFormation console with the URL <https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/events?stackId=arn%3Aaws%3Acloudformation%3Aus-east-1%3A324783324460%3Astac...>. The 'Events' tab is selected, displaying 20 events. The first event is for the stack creation, showing a timestamp of 2025-08-16 01:00:11 UTC-0400, logical ID 'demostack', status 'CREATE_COMPLETE', and detailed status '-'. Subsequent events show the creation of various resources like rdsDB, rdsSG, and rdsSubnetGroup, each with their respective timestamps, logical IDs, statuses, and detailed statuses. There are annotations: one orange arrow points to the 'Events' tab, and another points to the 'Timeline view' button at the top right of the events table.

Click on “Timeline View”

Prepared by Sidney Smith Ebot

The screenshot shows the AWS CloudFormation console with the 'demostack' stack selected. The 'Events' tab is active, displaying the 'Latest deployment timeline - new'. The timeline shows the creation of various resources over time. The resources listed are rdsDB, rdsSubnetGroup, PrivateSubnet2, rdsSG, PrivateSubnet1, and rdsVPC. The status for each resource is 'Complete'. The timeline spans from Aug 16, 00:54:00 to Aug 16, 01:00:00.

Resource	Time	Status
rdsDB	Aug 16, 00:54:00 - 01:00:00	Complete
rdsSubnetGroup	Aug 16, 00:54:30 - 00:55:00	Complete
PrivateSubnet2	Aug 16, 00:55:30 - 00:56:00	Complete
rdsSG	Aug 16, 00:56:30 - 00:57:00	Complete
PrivateSubnet1	Aug 16, 00:57:30 - 00:58:00	Complete
rdsVPC	Aug 16, 00:58:30 - 00:59:00	Complete

Let us check if our resources have been created. Go to VPC dashboard on AWS Console

The screenshot shows the AWS VPC dashboard with the 'Your VPCs' section selected. It lists three VPCs: rdsVPC, - (which is a placeholder), and Dev-VPN. All three VPCs are in an 'Available' state with 'Off' for 'Block Public...'. Their respective IPv4 CIDRs are 192.168.0.0/16, 172.31.0.0/16, and 10.100.0.0/16. The last update was 1 minute ago.

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP opt
rdsVPC	vpc-0f9eabf583e05544e	Available	Off	192.168.0.0/16	-	dopt-06
-	vpc-0128e9209eaef1c37	Available	Off	172.31.0.0/16	-	dopt-06
Dev-VPN	vpc-0c565a10e97a2bb88	Available	Off	10.100.0.0/16	-	dopt-06

You can see that the VPC has been created. Also go to the Database dashboard on AWS Management console

Prepared by Sidney Smith Ebot

The screenshot shows the AWS RDS Databases console. On the left, there's a sidebar with options like Aurora and RDS, Databases, Query editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Zero-ETL integrations, Events, Event subscriptions, Recommendations (0), and Certificate update. The main area is titled "Databases (1)" and shows a table with one row. The row details are: DB identifier (myrdsinstance), Status (Available), Role (Instance), Engine (MySQL Community), Region (us-east-1b), and Size (db.t5.micro). There are buttons for Group resources, Modify, Actions, and Create database.

You can see that our Database has been created
Also check the S3 dashboard on AWS Management Console

The screenshot shows the AWS S3 General purpose buckets page. On the left, there's a sidebar with options like General purpose buckets, Directory buckets, Table buckets, Vector buckets, Access Grants, Access Points (General Purpose Buckets, FSx file systems), Access Points (Directory Buckets), Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, Storage Lens (Dashboards, Storage Lens groups, AWS Organizations settings), Feature spotlight (11), and AWS Marketplace for S3. The main area shows a table for "General purpose buckets (1)". The bucket details are: Name (cf-templates-10oyihm9dl3s-us-east-1), AWS Region (US East (N. Virginia) us-east-1), and Creation date (August 15, 2025, 23:49:55 (UTC-04:00)). There are buttons for Copy ARN, Empty, Delete, and Create bucket. To the right, there are two callout boxes: "Account snapshot" (updated daily, View dashboard) which says "Storage Lens provides visibility into storage usage and activity trends.", and "External access summary - new" (updated daily) which says "External access findings help you identify bucket permissions that allow public access or access from other AWS accounts."

You can see that an S3 bucket has also been created.