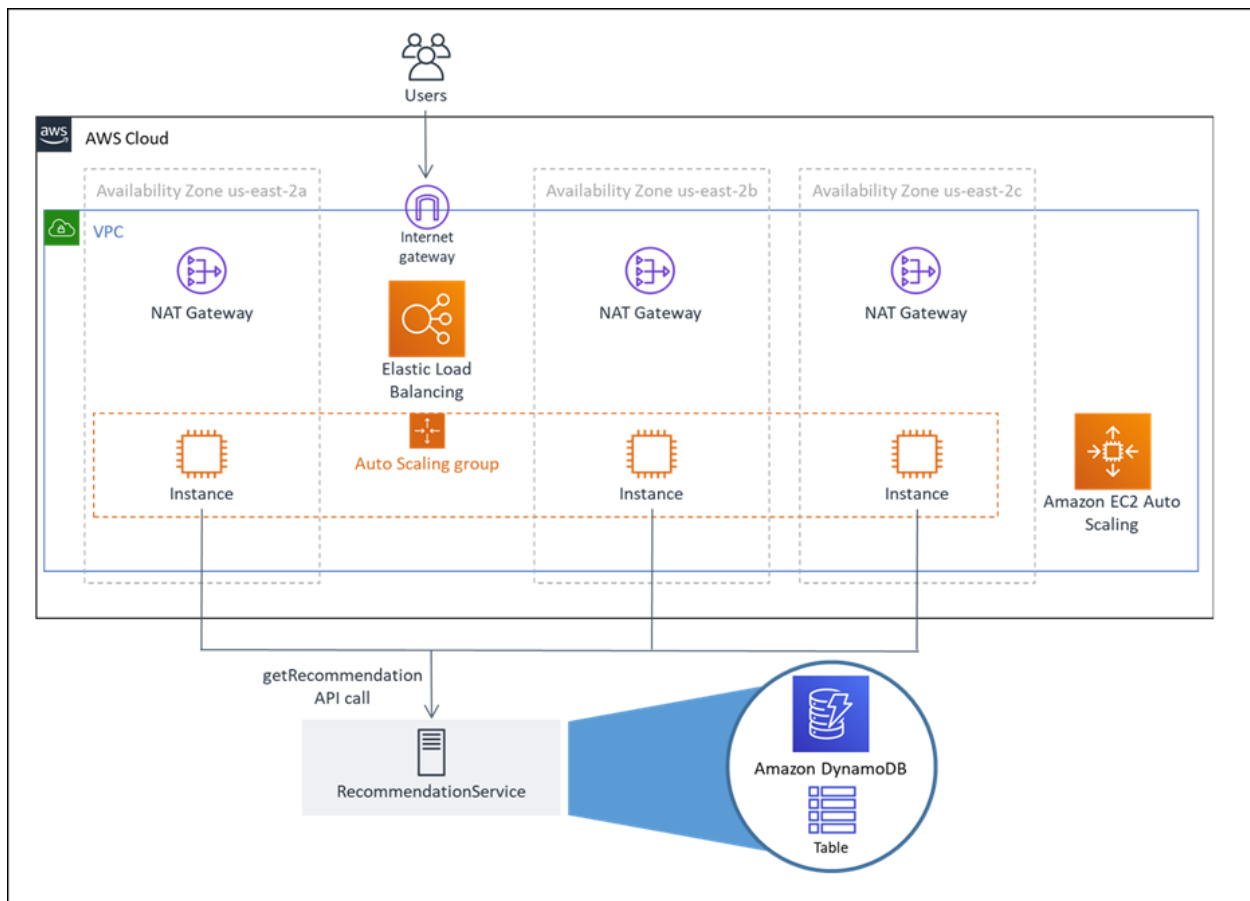


# Deploy a Reliable Multi-tier Infrastructure using CloudFormation

## Introduction

This hands-on lab will guide you through the steps to improve reliability of a service by using automation to deploy a reliable cloud infrastructure. When this lab is completed, you will have deployed two CloudFormation templates. The first will deploy an Amazon Virtual Private Cloud (VPC). The second will deploy into your VPC, a reliable 3-tier infrastructure using Amazon EC2 distributed across three Availability Zones. You will then review the features of the deployed infrastructure and learn how they contribute to reliability.

The architecture of the infrastructure you will deploy is represented by this diagram:



## Goals

By the end of this lab, you will be able to:

- Automate infrastructure deployment for a workload
- Understand how the deployed workload infrastructure contributes to reliability of the workload

## Steps

Deploy VPC using CloudFormation

Examine VPC resources created using CloudFormation

Deploy Web Application and Infrastructure using CloudFormation

Explore the Web Application

Explore the CloudFormation Template

Tear down this lab

# Deploy VPC using CloudFormation

## 1.1 Log into the AWS console


1. Login to AWS Console using AWS Event access link provided.

**Terms & Conditions:**

1. By using the Event Engine for the relevant event, you agree to the [AWS Event Terms and Conditions](#) and the [AWS Acceptable Use Policy](#). You acknowledge and agree that are using an AWS-owned account that you can only access for the duration of the relevant event. If you find residual resources or materials in the AWS-owned account, you will make us aware and cease use of the account. AWS reserves the right to terminate the account and delete the contents at any time.
2. You will not: (a) process or run any operation on any data other than test data sets or lab-approved materials by AWS, and (b) copy, import, export or otherwise create derivate works of materials provided by AWS, including but not limited to, data sets.
3. AWS is under no obligation to enable the transmission of your materials through Event Engine and may, in its discretion, edit, block, refuse to post, or remove your materials at any time.
4. Your use of the Event Engine will comply with these terms and all applicable laws, and your access to Event Engine will immediately and automatically terminate if you do not comply with any of these terms or conditions.

This is the 12 or 16 digit hash that was given to you for this event or for a specific team.

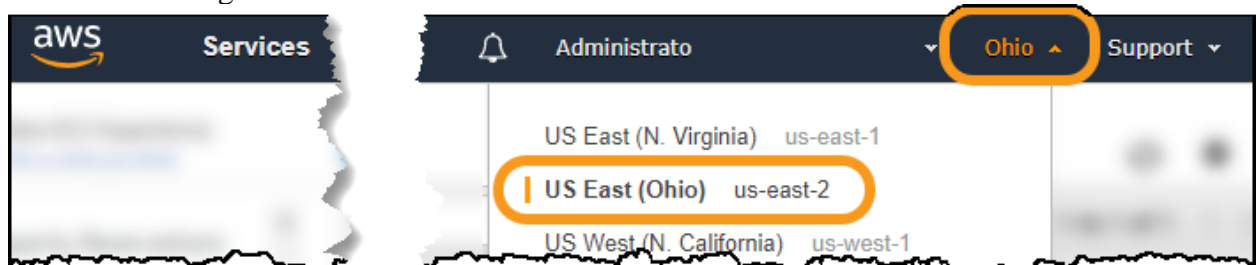
Click button



✓ Accept Terms & Login

## 1.2 Configure your AWS Region

1. Select the **Ohio** region. This region is also known as **us-east-2**, which you will see referenced throughout this lab.

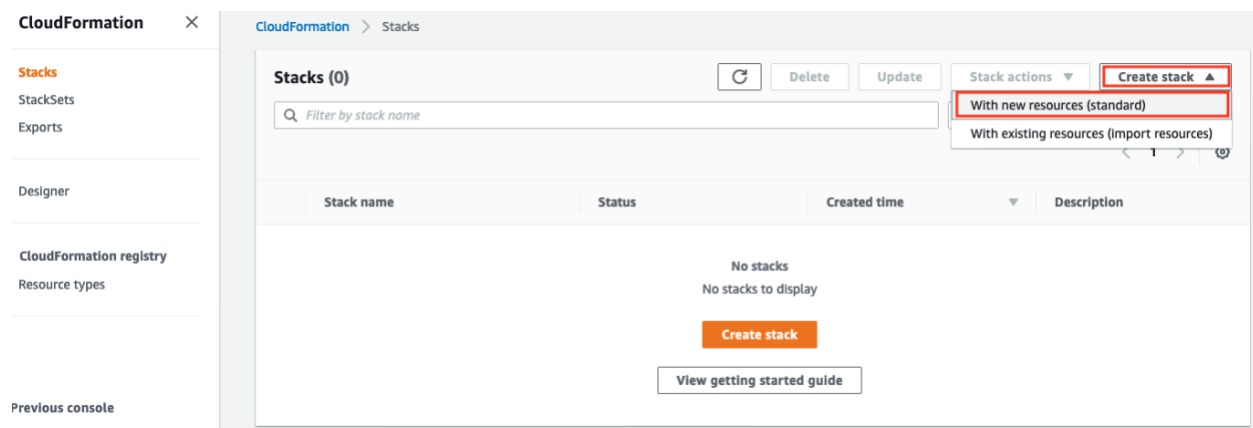


- AWS offers you the ability to deploy to over 20 regions located across the globe
- Each region is fully isolated from the others to isolate any issues and achieve high availability,
- Each region is comprised of multiple Availability Zones, which are fully isolated partitions of our infrastructure (more on this later)

## 1.3 Deploy the VPC infrastructure

This step will create the VPC and all components using the example CloudFormation template.

1. Download the latest version of the CloudFormation template here: [vpc-alb-app-db.yaml](https://vpc-alb-app-db.yaml)
2. Open the CloudFormation console at <https://console.aws.amazon.com/cloudformation/>
3. Click **Create Stack**, then **With new resources (standard)**.



4. Click **Upload a template file** and then click **Choose file**.

## Create stack

### Prerequisite - Prepare template

#### Prepare template

Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

☒ Template is ready

☐ Use a sample template

☐ Create template in Designer

### Specify template

A template is a JSON or YAML file that describes your stack's resources and properties.


#### Template source

Selecting a template generates an Amazon S3 URL where it will be stored.

☐ Amazon S3 URL

☒ Upload a template file

#### Upload a template file

Choose file 

No file chosen

JSON or YAML formatted file

S3 URL: Will be generated when template file is uploaded

[View in Designer](#)

Cancel

Next

5. Choose the CloudFormation template you downloaded in step 1, return to the CloudFormation console page and click **Next**.
6. Enter the following details:
  - **Stack name:** The name of this stack. For this lab, use **WebApp1-VPC** and match the case.
  - **Parameters:** Parameters may be left as defaults; you can find out more in the description for each.

## Specify stack details

### Stack name

Enter stack name here

Stack name

WebApp1-VPC

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

7. At the bottom of the page click **Next**.
8. In this lab, we use tags, which are key-value pairs, that can help you identify your stacks. Enter *Owner* in the left column which is the key, and your email address in the right column which is the value. We will not use additional permissions or advanced options so click **Next**. For more information, see [Setting AWS CloudFormation Stack Options](#)

## Configure stack options

### Tags

You can specify tags (key-value pairs) to apply to resources in your stack. You can add up to 50 unique tags for each stack. [Learn more](#) 

Owner

email@domain.com

Remove

Add tag

- Review the information for the stack. When you're satisfied with the configuration, at the bottom of the page check **I acknowledge that AWS CloudFormation might create IAM resources with custom names** then click **Create stack**.

### Capabilities

#### The following resource(s) require capabilities: [AWS::IAM::Role]

This template contains Identity and Access Management (IAM) resources. Check that you want to create each of these resources and that they have the minimum required permissions. In addition, they have custom names. Check that the custom names are unique within your AWS account. [Learn more](#).



I acknowledge that AWS CloudFormation might create IAM resources with custom names.

Cancel

Previous

Create change set

Create stack

- Click on **Create stack** and you will notice your stack creation started with status **CREATE\_IN\_PROGRESS**. CloudFormation begins creating the resources that are specified in the template.

### 1.4 Monitor CloudFormation Stack Creation

- Explore the **Events** tab where you can see progress as your stack get created. Click the refresh icon to the right to view updates. The Events tab shows detailed progress of all

resources being created as part of the template.

## WebApp1-VPC

DeleteUpdate

Stack actions ▾

Create stack ▾

Stack info

Events

Resources

Outputs

Parameters

Template

Change sets

Events (100+)

Timestamp ▾	Logical ID	Status	Status reason
2021-07-28 15:20:46 UTC-0700	ALB1Subnet3RouteToInternet	✔ CREATE_COMPLETE	-
2021-07-28 15:20:45 UTC-0700	DB1SubnetNetworkAclAssociation1	✔ CREATE_COMPLETE	-
2021-07-28 15:20:45 UTC-0700	ALB1SubnetNetworkAclAssociation3	✔ CREATE_COMPLETE	-
2021-07-28 15:20:45 UTC-0700	DB1SubnetRouteTableAssociation3	✔ CREATE_COMPLETE	-
2021-07-28 15:20:45 UTC-0700	DB1SubnetNetworkAclAssociation3	✔ CREATE_COMPLETE	-
2021-07-28 15:20:44 UTC-0700	Shared1SubnetRouteTableAssociation3	✔ CREATE_COMPLETE	-
2021-07-28 15:20:44 UTC-0700	DB1SubnetRouteTableAssociation1	✔ CREATE_COMPLETE	-
2021-07-28 15:20:44 UTC-0700	App1SubnetRouteTableAssociation2	✔ CREATE_COMPLETE	-
2021-07-28 15:20:44 UTC-0700	App1SubnetRouteTableAssociation3	✔ CREATE_COMPLETE	-
2021-07-28 15:20:44 UTC-0700	Shared1SubnetRouteTableAssociation2	✔ CREATE_COMPLETE	-

2. While you are waiting, explore all other tabs like the **Template** tab to review your template and the **Parameters** tab to see parameter values. You will examine the Template in more detail later in the lab.
3. Click on Stack Info tab to view the Overall status for the stack. Once stack status changes to **CREATE\_COMPLETE**, you can proceed to next step.

WebApp1-VPC

Delete

Update

Stack actions ▼

Create stack ▼

Stack info

Events

Resources

Outputs

Parameters

Template

Change sets

Overview

Stack ID

arn:aws:cloudformation:us-east-2:515903645640:stack/WebApp1-VPC/dabf9ec0-eff1-11eb-9bfd-069e79c6fc2e

Description

This template is from: <https://www.wellarchitectedlabs.com/> AWS CloudFormation Sample Template for VPC. Creates a multi-tier network appropriate for a web application including subnets for application, application load balancer (can disable), database (can disabled using parameter), and shared services (can disable). **\*\*WARNING\*\*** You will be billed for the AWS resources created if you create a stack from this template. Copyright 2019-2020 Amazon.com, Inc. or its affiliates. All Rights Reserved. Licensed under the Apache License, Version 2.0 (the "License"). You may not use this file except in compliance with the License. A copy of the License is located at <https://www.apache.org/licenses/LICENSE-2.0> or in the "license" file accompanying this file. This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Status

✔ CREATE\_COMPLETE

Status reason

-

- Once stack creation is complete, visit the **Resources** tab to see all the resources that were created by this AWS CloudFormation template.

CloudFormation > Stacks > WebApp1-VPC

Stacks

WebApp1-VPC

Delete

Update

Stack actions ▼

Create stack ▼

Stack info

Events

Resources

Outputs

Parameters

Template

Change sets

Resources (142)

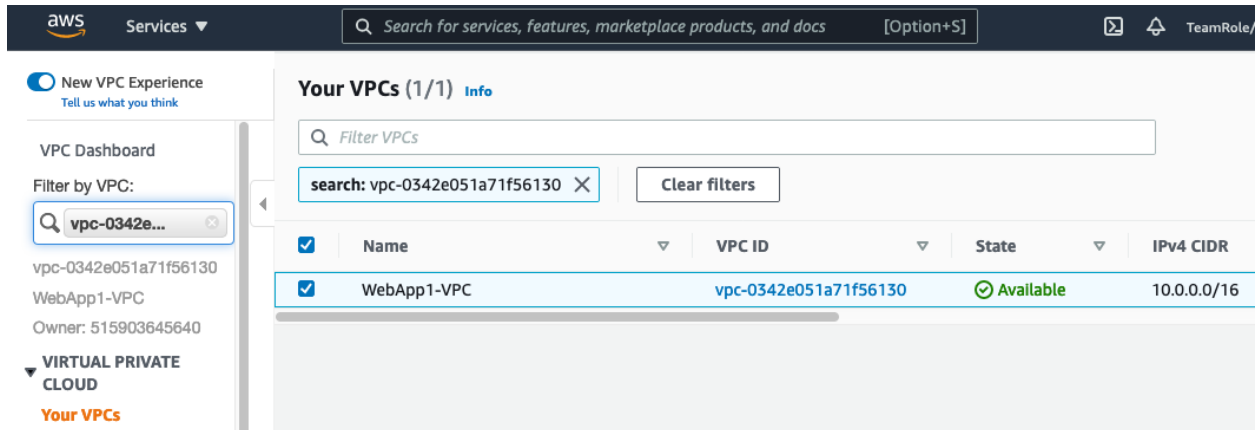
Logical ID	Physical ID	Type	Status	Status reason
ALB1InFromInternetHTTPAclEntry	WebAp-ALB1I-1MK12C61BAYL3	AWS::EC2::NetworkAclEntry	✔ CREATE_COMPLETE	-
ALB1InFromInternetHTTPAclEntryv6	WebAp-ALB1I-1QCVMZMKA8I93	AWS::EC2::NetworkAclEntry	✔ CREATE_COMPLETE	-
ALB1InFromInternetHTTPSaclEntry	WebAp-ALB1I-1TEFLP6W7HNEL	AWS::EC2::NetworkAclEntry	✔ CREATE_COMPLETE	-
ALB1InFromInternetHTTPSaclEntryv6	WebAp-ALB1I-1HA9EKUGWTHF2	AWS::EC2::NetworkAclEntry	✔ CREATE_COMPLETE	-
ALB1InNetworkEphemeralVPAAclEntry1	WebAp-ALB1I-1CBUDT5INDNBY	AWS::EC2::NetworkAclEntry	✔ CREATE_COMPLETE	-
ALB1InNetworkEphemeralVPAAclEntry1v6	WebAp-ALB1I-MS6QXI6WCSQ	AWS::EC2::NetworkAclEntry	✔ CREATE_COMPLETE	-

## Examine VPC resources created using CloudFormation

In this task, you will examine a few of the VPC resources that were created together with the code from the CloudFormation template that created the resources.

- In the AWS Console search bar, Search for and Select **VPC**

In **Filter by VPC** in the top-left corner, select the VPC with Name tag: **WebApp1-VPC**. Click on **Your VPCs**.



Here is the code from the CloudFormation template that created this VPC:

Resources:

```
VPC:
  Type: 'AWS::EC2::VPC'
  Properties:
    CidrBlock: !Ref VPCIPv4CidrBlock
    EnableDnsSupport: true
    EnableDnsHostnames: true
    InstanceTenancy: default
    Tags:
      - Key: Name
        Value: !Join
          - '-'
          - - !Ref NamingPrefix
            - VPC
```

The **Type** parameter in the above code declares the type of resources being created by CloudFormation.

The **Properties** section specifies more information about the resource to create. In this case, it defines:

- **CidrBlock:** The IP address range associated with the VPC.
- **EnableDnsSupport:** Configures the VPC to use Route 53 to resolve DNS hostnames
- **EnableDnsHostnames:** Configures the VPC to associate DNS names with Amazon EC2 instances.
- **Tags:** Adds a friendly name to the resource.



The **CidrBlock** property is defined by using the intrinsic function `Ref`, which in this example returns the value of the *parameter* `<VPCIPv4CidrBlock>` defined in the Parameters section of the CloudFormation template

```
Parameters:
<...>
  VPCIPv4CidrBlock:
    Description: VPC CIDR block for IPv4. Default of 10.0.0.0/16 is
    recommended for testing.
    Type: String
    Default: 10.0.0.0/16
<..>
```

The **Tag** Parameter assigns a Name Tag to the VPC, which uses the `!Join` function to assign a name to the VPC: **WebApp1-VPC**

2. In the left navigation pane for the VPC Dashboard, click **Internet Gateways**. The Internet Gateway created using CloudFormation and all other created resources use the same naming convention. `<WebApp1>`

Click the checkbox to the left of the WebApp1-IGW Name and navigate to the **Tags** Tab at the bottom.

Here you will find additional Tags that were applied to identify resources created using the CloudFormation template

The screenshot shows the AWS VPC Dashboard. In the left navigation pane, 'Internet Gateways' is selected. The main panel displays a table of Internet Gateways. The first gateway, 'WebApp1-IGW', is selected. Below the table, the 'Tags' tab is active, showing a single tag: 'WebApp1-VPC'.

Name	Internet gateway ID	State	VPC ID	Owner
<input checked="" type="checkbox"/> WebApp1-IGW	igw-0fdb41fef610c83da	Attached	vpc-055d30c2cf3825c5f   WebApp1-VPC	515903645640
<input type="checkbox"/> -	igw-5eb1b336	Attached	vpc-4ae38321	515903645640

igw-0fdb41fef610c83da / WebApp1-IGW		
Details		
Internet gateway ID	State	VPC ID
igw-0fdb41fef610c83da	Attached	vpc-055d30c2cf3825c5f   WebApp1-VPC

igw-0fdb41fef610c83da / WebApp1-IGW	
Details Tags	
Tags	
Q Search tags	
Key	Value
aws:cloudformation:stack-id	arn:aws:cloudformation:us-east-2:515903645640:stack/WebApp1-VPC/dabf9ec0-eff1-11eb-9bfd-069e79c6fc2e
aws:cloudformation:stack-name	WebApp1-VPC
aws:cloudformation:logical-id	IGW
Owner	
Name	WebApp1-IGW

Below is the code from the CloudFormation template that created this IGW and attached it to **WebApp1-VPC**:

```
IGW:
  Type: 'AWS::EC2::InternetGateway'
  Properties:
    Tags:
      - Key: Name
        Value: !Join
          - '-'
          - - !Ref NamingPrefix
            - IGW
IGWAttach:
  Type: 'AWS::EC2::VPCGatewayAttachment'
  Properties:
    VpcId: !Ref VPC
    InternetGatewayId: !Ref IGW
```

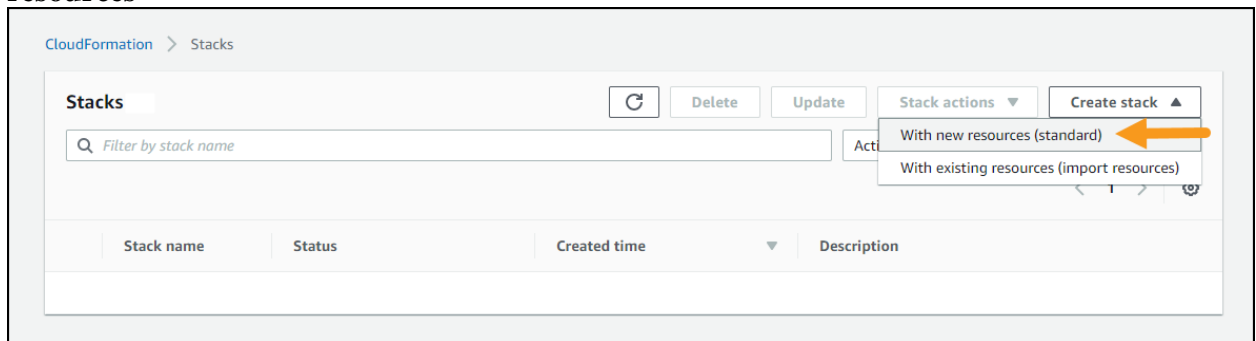
The `IGWAttach` code snippet also uses the `!Ref` function. In this example, the `!Ref` function is used to reference other resources, the VPC and IGW, that were created within the template.

## Deploy Web Application and Infrastructure using CloudFormation

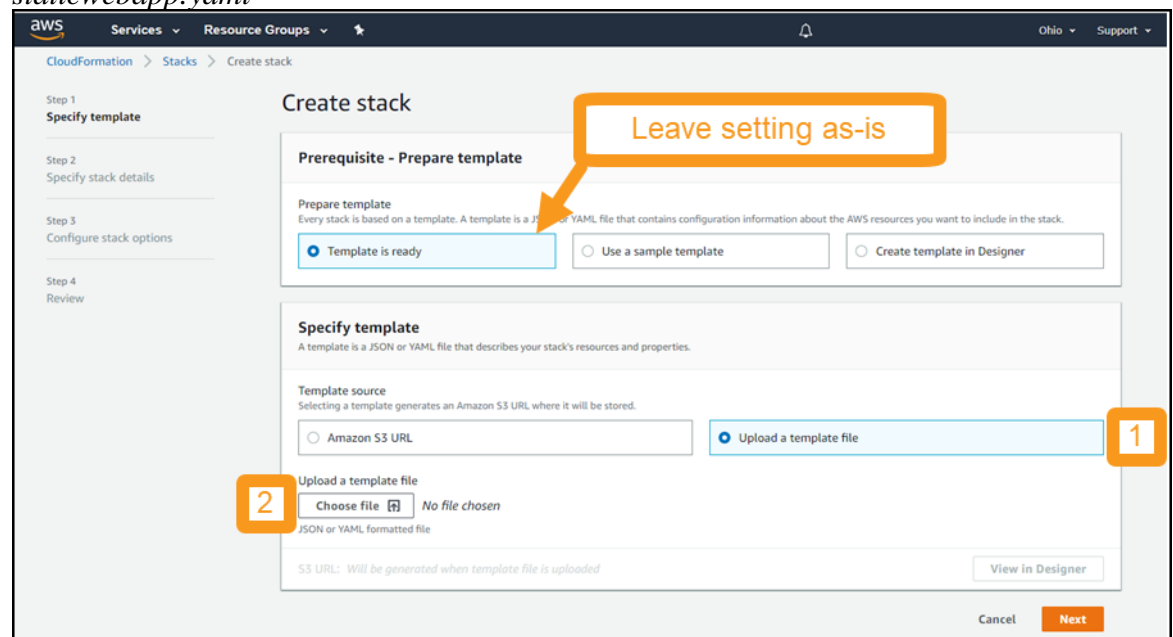
Download the CloudFormation template that will be used to deploy a Web Application in the VPC that was created in the first lab: [staticwebapp.yaml](#)

1. Go to the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation> and click **Create Stack > With new**

## resources



2. Leave **Prepare template** setting as-is
  - For **Template source** select **Upload a template file**
  - Click **Choose file** and supply the CloudFormation template you downloaded:  
*staticwebapp.yaml*



3. Click **Next**
4. For **Stack name** use **CloudFormationLab**
5. **Parameters**
  - Look over the Parameters and their default values.
  - Click **Next**
6. For **Configure stack options** we recommend configuring tags, which are key-value pairs, that can help you identify your stacks and the resources they create. For example, enter *Owner* in the left column which is the key, and your email address in the right column which is the value. We will not use additional permissions or advanced options so click **Next**
7. For **Review**
  - Review the contents of the page
  - At the bottom of the page, select **I acknowledge that AWS CloudFormation might create IAM resources with custom names**

- Click **Create stack**

Capabilities

**The following resource(s) require capabilities: [AWS::IAM::Role]**

This template contains Identity and Access Management (IAM) resources. Check that you want to create each of these resources and that they have the minimum required permissions. In addition, they have custom names. Check that the custom names are unique within your AWS account. [Learn more.](#)

☒ I acknowledge that AWS CloudFormation might create IAM resources with custom names.

Cancel Previous Create change set **Create stack**

- This will take you to the CloudFormation stack status page, showing the stack creation in progress.
  - Click on the **Events** tab
  - Scroll through the listing. It shows the activities performed by CloudFormation (newest events at top), such as starting to create a resource and then completing the resource creation.
  - Any errors encountered during the creation of the stack will be listed in this tab.

CloudFormation > Stacks > HealthCheckLab

Stacks (2)

Active ☐ View nested < 1 >

HealthCheckLab  
2020-02-26 11:06:19 UTC-0800  
**CREATE\_IN\_PROGRESS**

WebApp1-VPC  
2020-02-25 20:10:26 UTC-0800  
**CREATE\_COMPLETE**

**Stack Status**

**Events (37)**

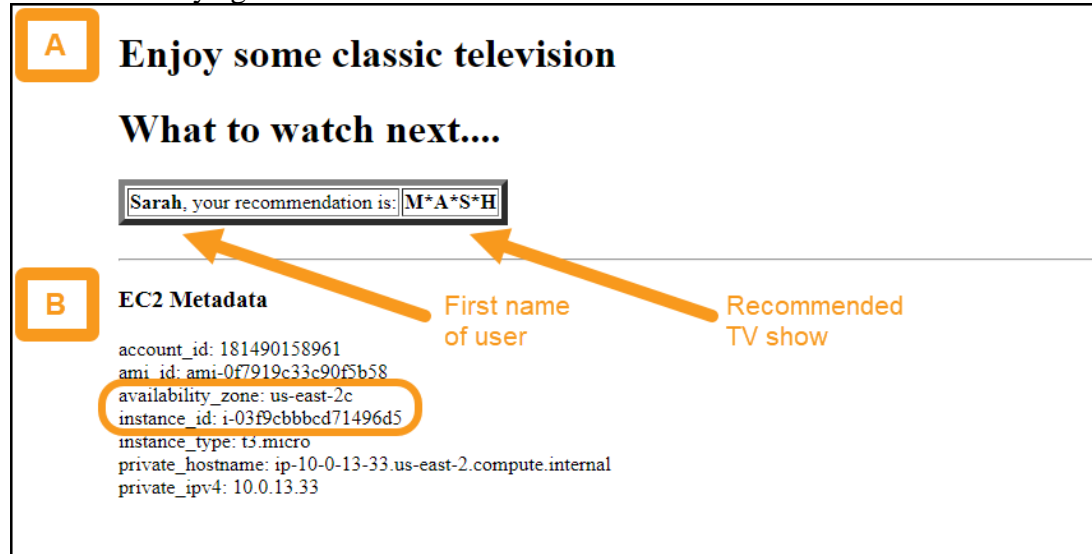
Timestamp	Logical ID	Status	Status reason	Type
2020-02-26 11:07:53 UTC-0800	Web1InstanceIn stanceProfile	<b>CREATE_IN_PROGRESS</b>	Resource creation Initiated	AWS::IAM::Ins tanceProfile
2020-02-26 11:07:53 UTC-0800	Web1InstanceIn stanceProfile	<b>CREATE_IN_PROGRESS</b>	-	AWS::IAM::Ins tanceProfile
2020-02-26 11:07:52 UTC-0800	Web2InstanceIn stanceProfile	<b>CREATE_IN_PROGRESS</b>	Resource creation Initiated	AWS::IAM::Ins tanceProfile
2020-02-26	Web2InstanceIn	<b>CREATE_IN_PROGRESS</b>	-	AWS::IAM::Ins

- When it shows **status CREATE\_COMPLETE**, then you are finished with this step. This should take about four minutes.

## Explore the Web Application

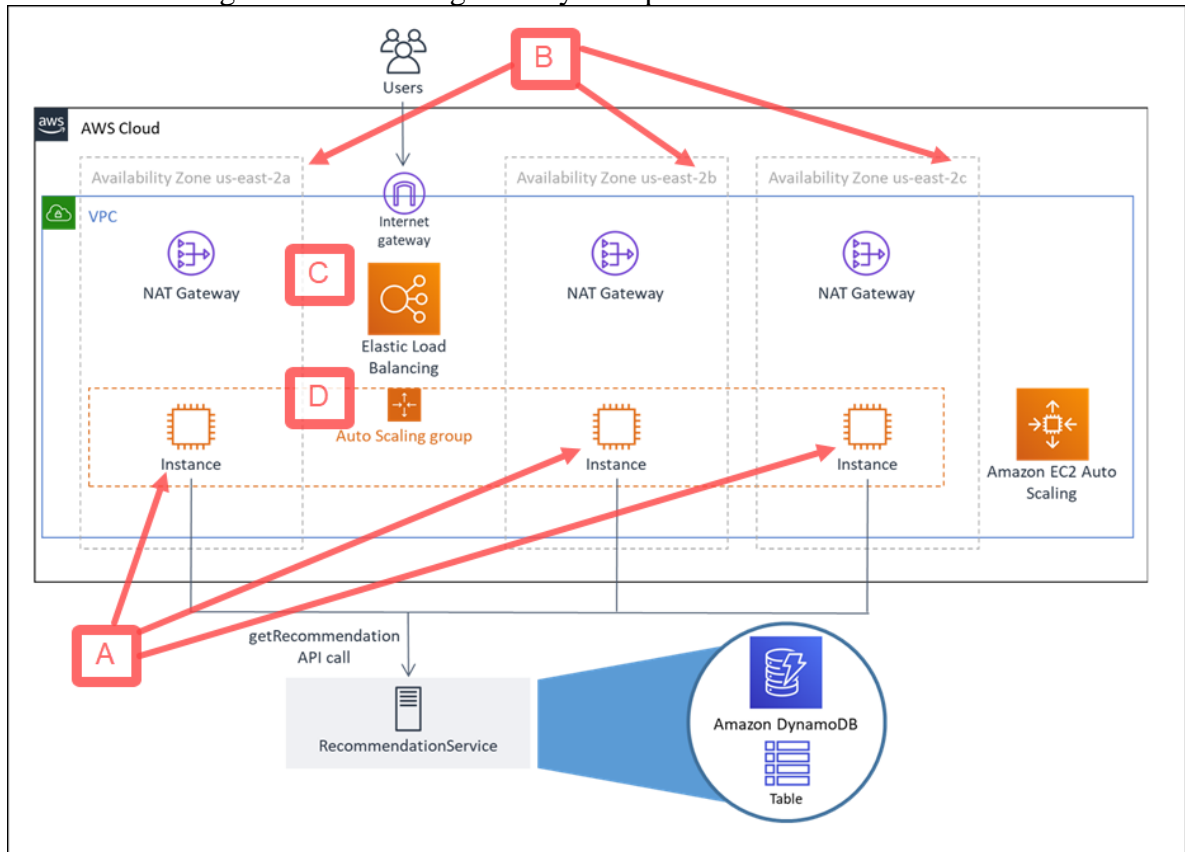
- Go to the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>.
  - Click on the **CloudFormationLab** stack
  - Click on the **Outputs** tab
  - For the Key **WebsiteURL** copy the value. This is the URL of your test web service

- *Hint:* it will start with `http://health-alb` and end in `<aws region>.elb.amazonaws.com`
- 2. Click the URL and it will bring up the website:
  - *Troubleshooting:* if you see an error such as *502 Bad Gateway*, then wait 60 seconds and try again. It takes some time for the servers to initialize.



3. The website simulates a recommendation engine making personalized suggestions for classic television shows. You should note the following features:
  - Area A shows the personalized recommendation
    - It shows first name of the user and the show that was recommended
    - The workshop simulation is simple. On every request it chooses a user at random, and shows a recommendation statically mapped to that user. The user names, television show names, and this mapping are in a DynamoDB table, which is simulating the **RecommendationService**
  - Area B shows metadata which is useful to you during the lab
    - The **instance\_id** and **availability\_zone** enable you to see which EC2 server and Availability Zone were used for each request

4. Use the following architectural diagram as you explore the site



- **A** - There is one EC2 instance deployed per Availability Zone
- **B** - Refresh the website several times, note that the EC2 instance and Availability Zone change from among the three available
- **C** - Elastic Load Balancing (ELB) is used here. An Application Load Balancer receives each request and distributes it among the available EC2 server instances across Availability Zones.
  - The requests are stateless, and therefore can be routed to any of the available EC2 instances
- **D** - The EC2 instances are in an [Amazon EC2 Auto Scaling Group](#). This Auto Scaling Group was configured to maintain three instances, therefore if one instance is detected as *unhealthy* it will be replaced to maintain three *healthy* instances.
  - AWS Auto Scaling can also be configured to scale up/down dynamically in response to workload conditions such as CPU utilization or request count.

**The Architecture deployed with the provided CloudFormation templates follows AWS Well-Architected Best Practices for Reliability**

**Use highly available network connectivity for your workload public endpoints:**

You used Elastic Load Balancing which provides load balancing across Availability Zones, performs Layer 4 (TCP) or Layer 7 (http/https) routing, integrates with AWS WAF, and integrates with AWS Auto Scaling to help create a self-healing infrastructure and absorb increases in traffic while releasing resources when traffic decreases.

**Implement loosely coupled dependencies:**

Dependencies such as load balancers are loosely coupled. Loose coupling helps isolate behavior of a component from other components that depend on it, increasing resiliency and agility.

**Deploy the workload to multiple locations:**

Distribute workload data and resources across multiple Availability Zones or, where necessary, across AWS Regions

**Automate healing on all layers:**

Upon detection of a failure, use automated capabilities to perform actions to remediate it

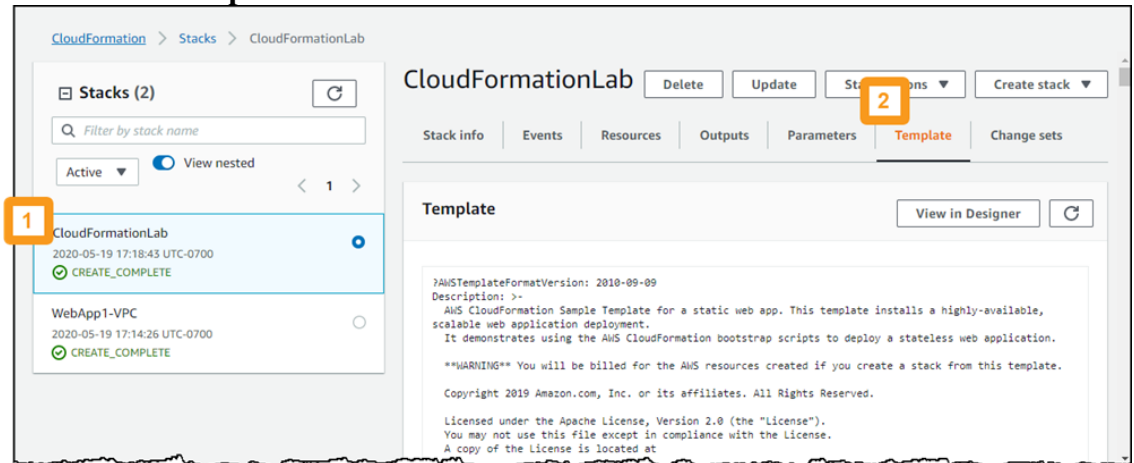
Additional best practices can be found in our [AWS Well-Architected Framework](#).

## **Explore the CloudFormation Template used to create the WebApp**

In this section you will explore the CloudFormation template and learn how you were able to deploy the web application infrastructure using it

- Go to the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>
  1. Click on the **CloudFormation** stack that you deployed

## 2. Click on the **Template** tab



- *Alternate:* You previously downloaded the CloudFormation Template *staticwebapp.yaml*. You can also view it in the text editor of your choice

The template is written in a format called [YAML](#) , which is commonly used for configuration files. CloudFormation templates can also be written in JSON.

Look through the template. You will notice several sections:

- The [Parameters section](#) is used to prompt for inputs that can be used elsewhere in the template. The template is asking for several inputs, but also provides default values for each one.
  - Look through these and start to reason about what some are used for.
  - For example, `InstanceType` is a parameter where the user can choose that Amazon EC2 instance type to deploy for the servers used in this Web App.
- The [Conditions section](#) is where you can setup *if/then*-like control of what happens during template deployment. It defines the circumstances under which entities are created or configured.
- The [Resources section](#) is the “heart” of the template. It is where you define the infrastructure to be deployed. Look at the *first* resource defined.
  - It is the Amazon DynamoDB table used as the mock for the **RecommendationService**
  - It has a *logical ID* which in this case is `DynamoDBServiceMockTable`. This logical ID is how we refer to the DynamoDB table resource within the CloudFormation template.
  - It has a `Type` which tells CloudFormation which type of resource to create. In this case a `AWS::DynamoDB::Table`
  - And it has `Properties` that define the values used to create the VPC
- The [Outputs section](#) is used to display selective information about resources in the stack.
  - In this case it uses the built-in function `!GetAtt` to get the DNS Name for the Application Load Balancer.
  - This URL is what you used to access the WebApp
- The [Metadata section](#) here is used to group and order how the CloudFormation parameters are displayed when you deploy the template using the AWS Console



# Tear down this lab

If you are attending an in-person workshop and were provided with an AWS account by the instructor:

- There is no need to tear down the lab. Feel free to continue exploring. Log out of your AWS account when done.

If you are using your own AWS account:

- You may leave these resources deployed for as long as you want. When you are ready to delete these resources, see the following instructions

How to delete an AWS CloudFormation stack

1. Go to the AWS CloudFormation console:  
<https://console.aws.amazon.com/cloudformation>
2. Select the **CloudFormationLab** stack and click **Delete**
3. Wait for the **CloudFormationLab** CloudFormation stack to complete (it will no longer be shown on the list of active stacks)
  - The **Status** changes to *DELETE\_IN\_PROGRESS*
  - Click the refresh button to update and status will ultimately progress to *DELETE\_COMPLETE*
4. Select the **WebApp1-VPC** stack and click **Delete**

## References & useful resources

AWS CloudFormation

- [What is AWS CloudFormation?](#)
- CloudFormation [AWS Resource and Property Types Reference](#)

AWS Resources that enable reliable architectures:

- [What Is Amazon EC2 Auto Scaling?](#)
- Elastic Load Balancing: [What Is an Application Load Balancer?](#)
- Availability Zones: AWS [Global Infrastructure](#)