Contenido

Guía Elaborada por: Juan Felipe Gómez Manzanares.
Ingeniero Telemático

Laboratorio – Introducción Cloud Formation

Objetivo Práctica Laboratorio

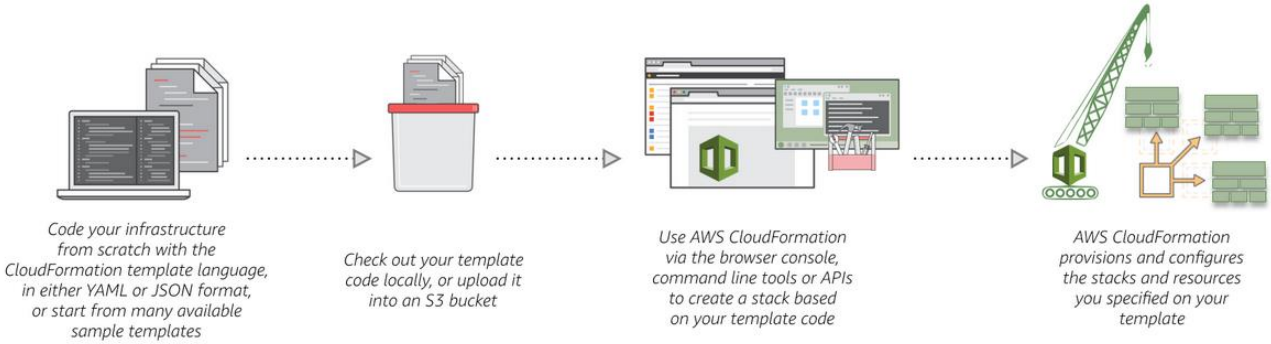
DESPLIEGUE infraestructura Amazon Web Services utilizando la herramienta Amazon Cloud Formation.

Cloud Formation

AWS CloudFormation |. CloudFormation permite usar un archivo de texto simple para modelar y aprovisionar, de una manera segura y automatizada, todos los recursos necesarios para las aplicaciones en todas las regiones y cuentas.

Dicho archivo puede ser un JSON o YAML, a través del cual, se describe una infraestructura AWS y su comportamiento. Con AWS CloudFormation es posible desarrollar Infraestructura cómo código (IaaC).

El diagrama 1, describe el funcionamiento general de AWS CloudFormation.

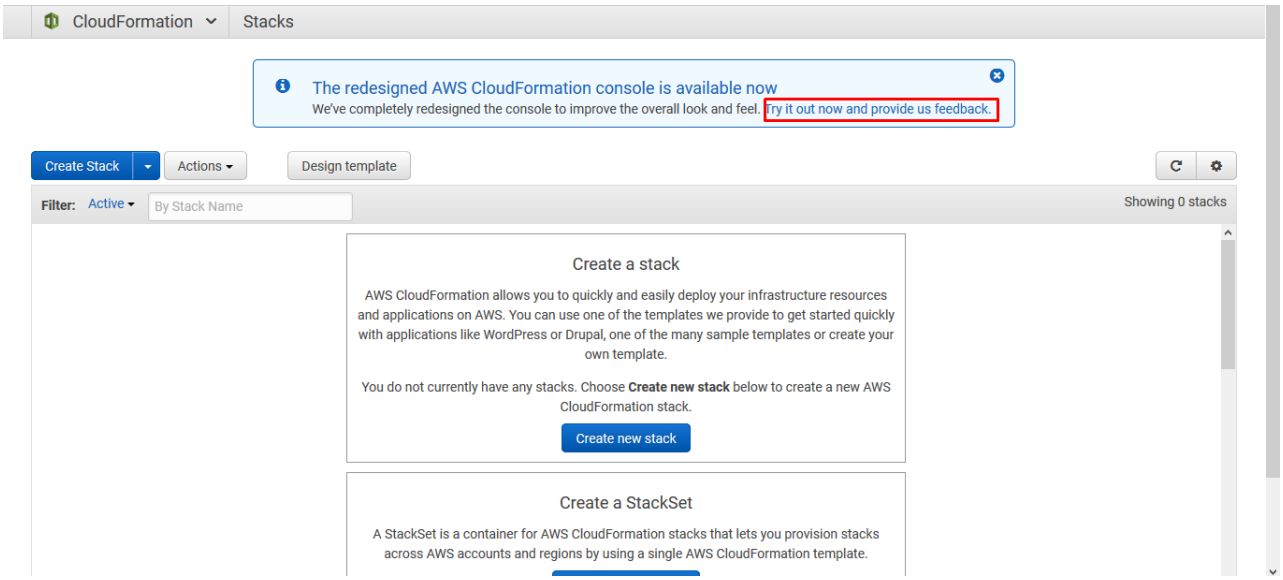


CloudFormation Design Tool – Añadir Recursos.

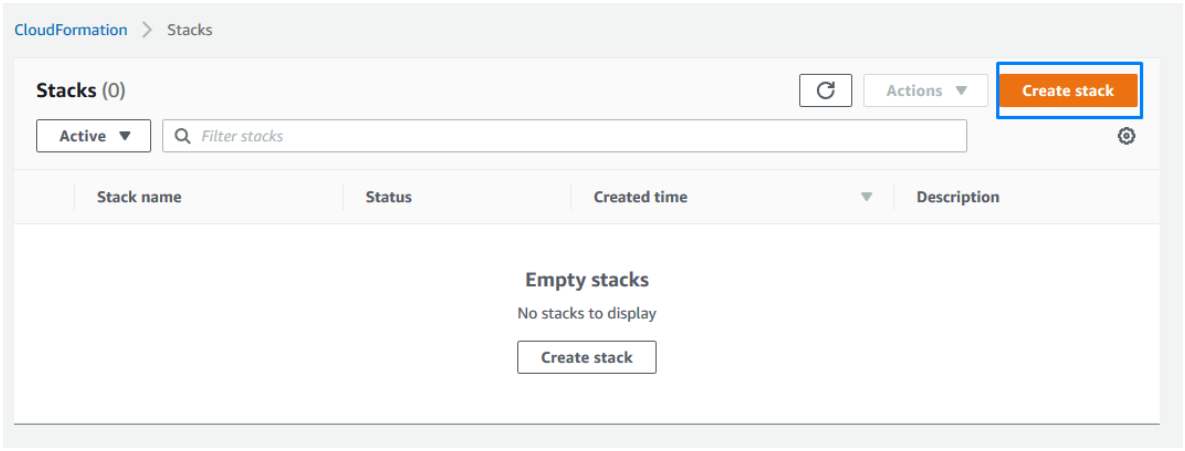
Crear Stack.

Lo primero es buscar el servicio de AWS CloudFormation dentro de los recursos de AWS. Una vez abierto el servicio, verá una ventana cómo la siguiente:

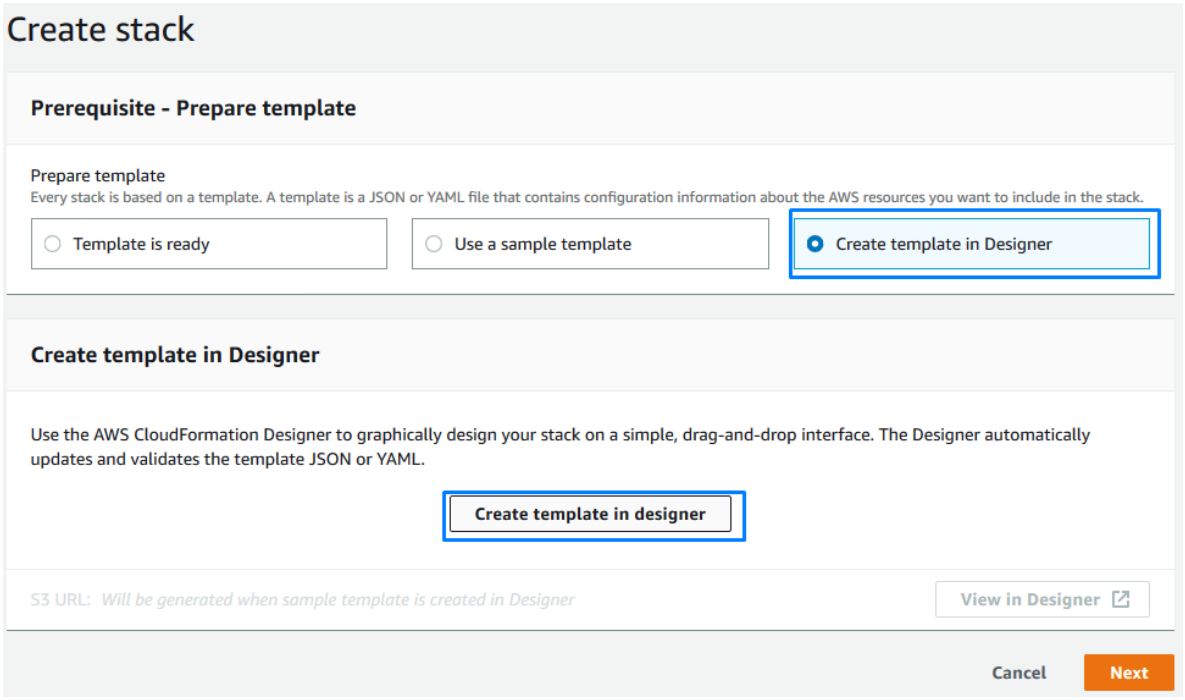
Guía Elaborada por: Juan Felipe Gómez Manzanares.
Ingeniero Telemático

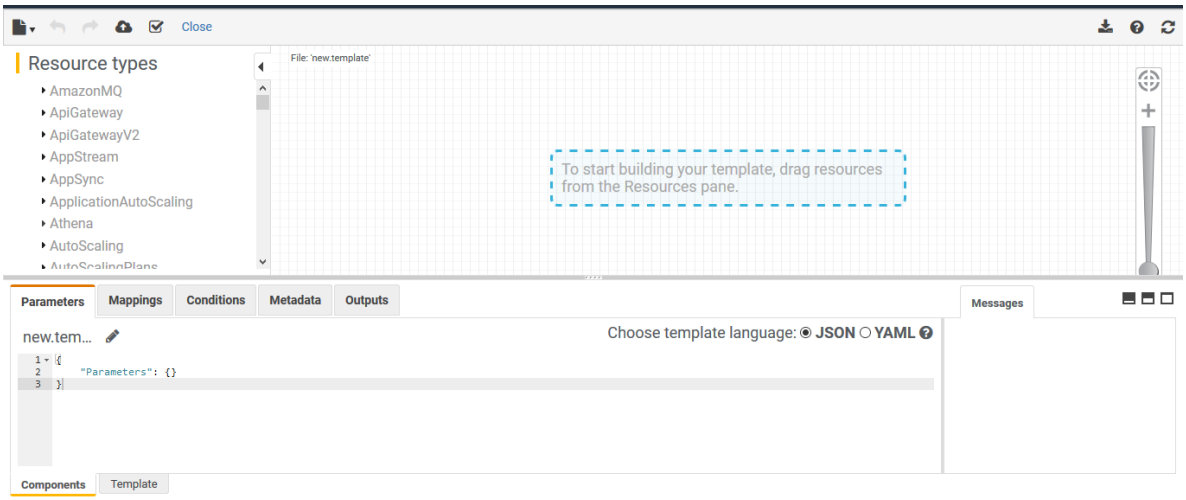


Se aconseja dar clic en la opción: "Try it out now and provide us feedback". Para trabajar con la interfaz actualizada. Una vez dentro, damos clic en la opción Create Stack, cómo se aprecia en la siguiente captura:



Ahora seleccionamos la opción: "Crear un Template in Designer". Está opción nos abrirá una nueva ventana a través de la cuál vamos a poder diseñar nuestra infraestructura AWS utilizando una aplicación web que nos permita insertar y conectar recursos de una manera muy simple.

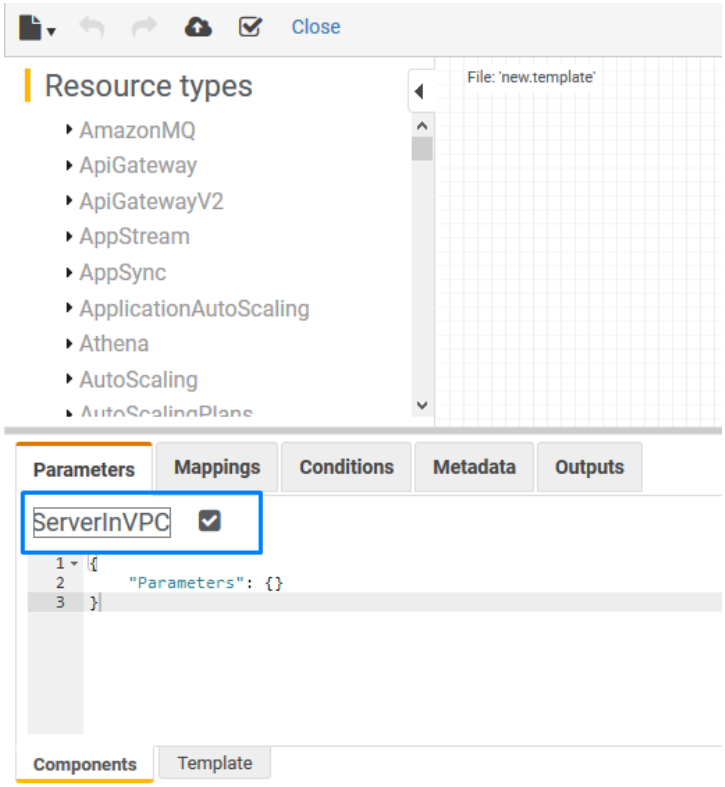




Con esta herramienta iremos creando paso a paso una plantilla que nos permita automatizar el proceso de creación de una infraestructura en AWS. Para iniciar, vamos a cambiar el nombre de la plantilla por **`BasicWebServerInVPC`**.

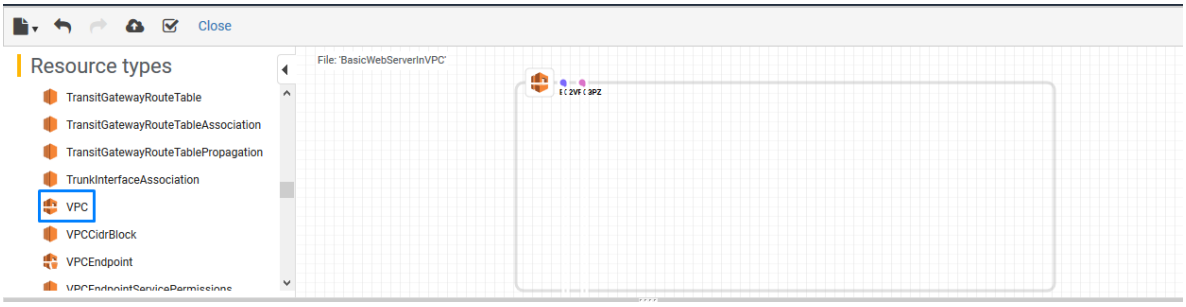
> Es importante crear los recursos con los mismos nombres para evitar fallos en la ejecución de la plantilla al final del laboratorio.

Guía Elaborada por: Juan Felipe Gómez Manzanares.
Ingeniero Telemático



Añadir VPC y recursos EC2.

Ahora vamos a empezar a agregar los recursos al lienzo en blanco. En el panel Resource types (Tipos de recursos), desde la categoría EC2, arrastre un tipo de recurso VPC hacia el panel Canvas (Lienzo). En este punto, usted debe haber notado que los recursos están organizados por categorías de recursos, y que se encuentran todos los posibles recursos de AWS.



AWS CloudFormation Designer modifica inmediatamente la plantilla para incluir un recurso de VPC, los resultados tienen un aspecto similar al siguiente fragmento de código JSON.

```
{
    "Resources": {
        "EC2VPC3PZ": {
            "Type": "AWS::EC2::VPC",
            "Properties": {}
        }
    }
}
```

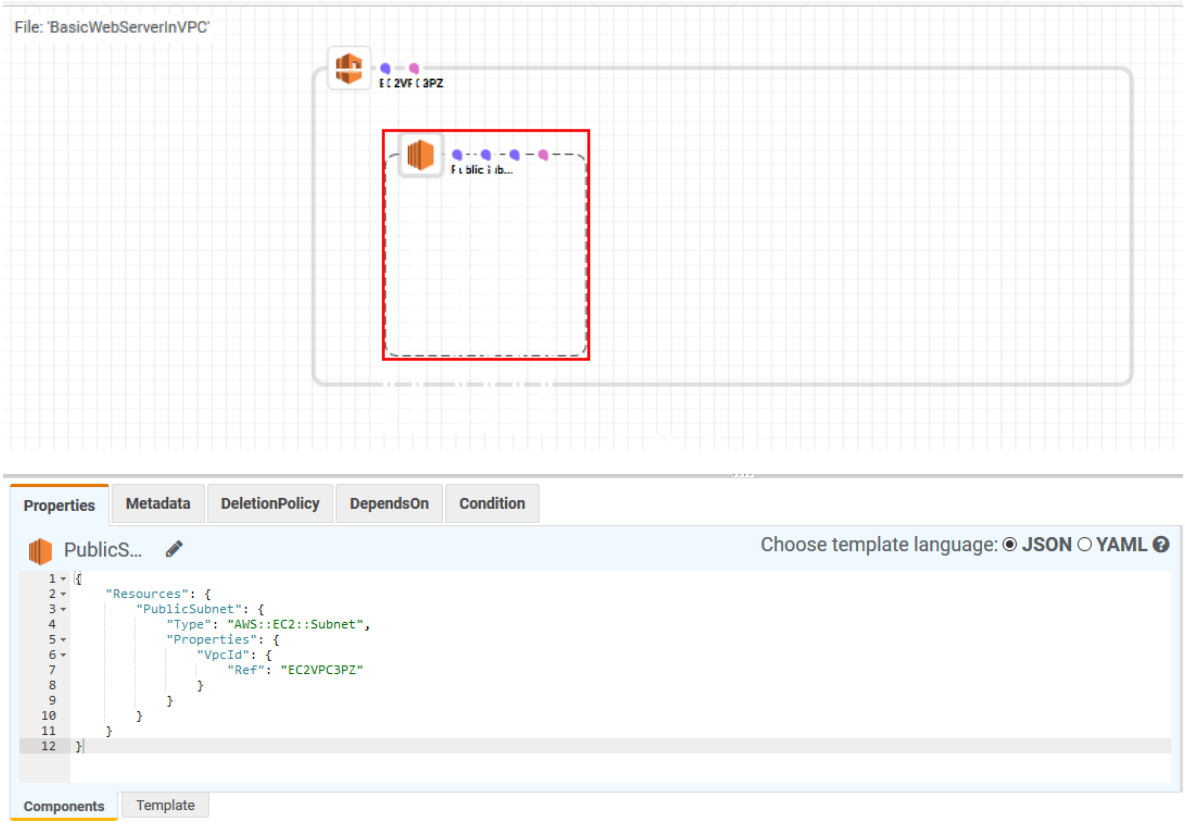
Adicionalmente, puede consultar la plantilla completa, presionando la pestaña template.

Ahora cambie el nombre de la VPC a **`VPC`**

Una vez cambiado el nombre, tenemos que añadir una **subred** para poder asociar una instancia EC2, que aloja el sitio web. Recuerde que las instancias deben estar en una subred. Añada un tipo de recurso Subnet dentro de la VPC y cámbiele el nombre por **`PublicSubnet`**.

Cuando añade la subred dentro de la VPC, AWS CloudFormation Designer asocia automáticamente la subred con la VPC. Esta asociación es un modelo de contenedor, donde los recursos dentro del contenedor se asocian automáticamente.

Al final de este punto debe tener la plantilla de esta manera:

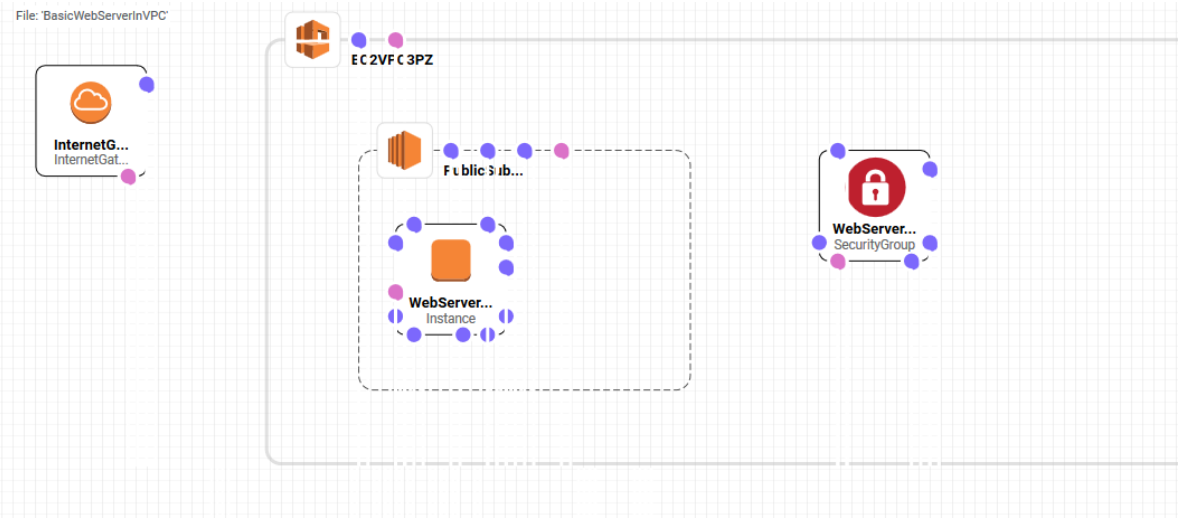


Añada un tipo de recurso **Instance** dentro del recurso **`PublicSubnet`** y cámbiele el nombre por **`WebServerInstance`**. De forma similar a la forma en que esto funcionó con la subred y la VPC, la adición de la instancia en la subred asocia automáticamente la instancia con la subred. Una vez modificado el nombre, la plantilla debe quedar de la siguiente manera:

Añada un tipo de recurso **SecurityGroup** dentro de la VPC y cámbiele el nombre por **`WebServerSecurityGroup`**. El grupo de seguridad es un firewall virtual que controla el tráfico entrante y saliente de la instancia del servidor web. También es necesario para las instancias de una VPC.

Adicionalmente, añada un tipo de recurso **InternetGateway** en cualquier lugar fuera de la VPC y cámbiele el nombre por **`InternetGateway`**. El Internet Gateway permite la comunicación entre la instancia que está dentro de la VPC e Internet. Sin este recurso, nadie puede obtener acceso a su sitio web.

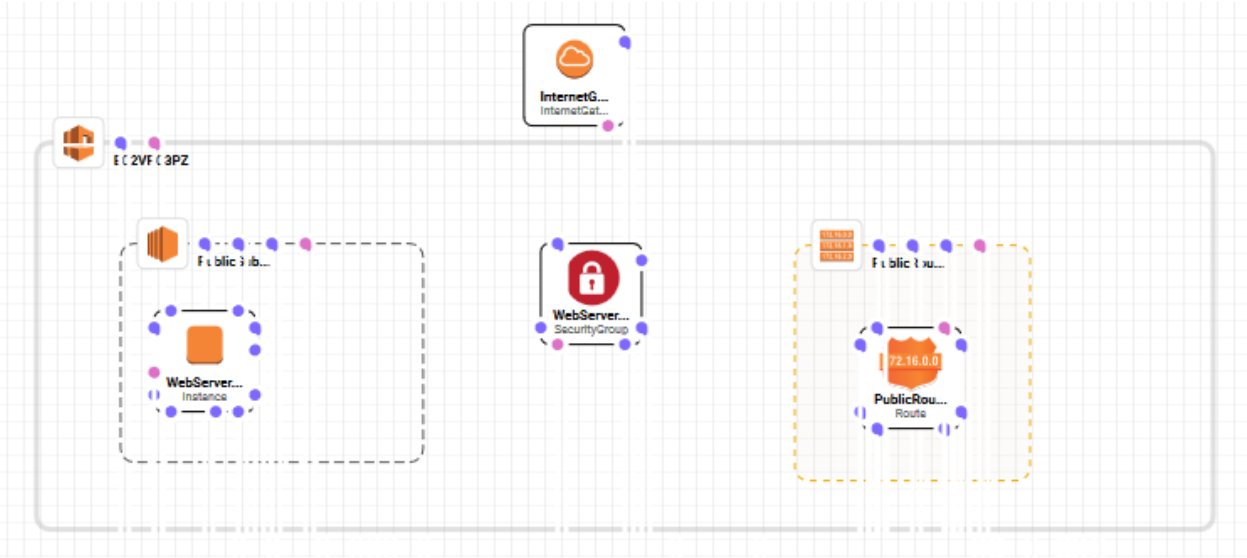
En este punto del laboratorio, deberá tener el siguiente diagrama en el canvas:



A continuación, debemos añadir una tabla de ruteo y una ruta para especificar cómo dirigir el tráfico de la red desde una subred. Añada una **RouteTable** dentro de la VPC y cámbiele el nombre por **`PublicRouteTable`**.

Para añadir una regla de enrutamiento en la tabla de ruteo, añada un tipo de recurso **Route** dentro del recurso PublicRouteTable y cámbiele el nombre por **`PublicRoute`**. Utilizaremos la ruta para especificar hacia dónde dirigir el tráfico.

En este punto, su diagrama de Canvas debe verse de la siguiente manera:

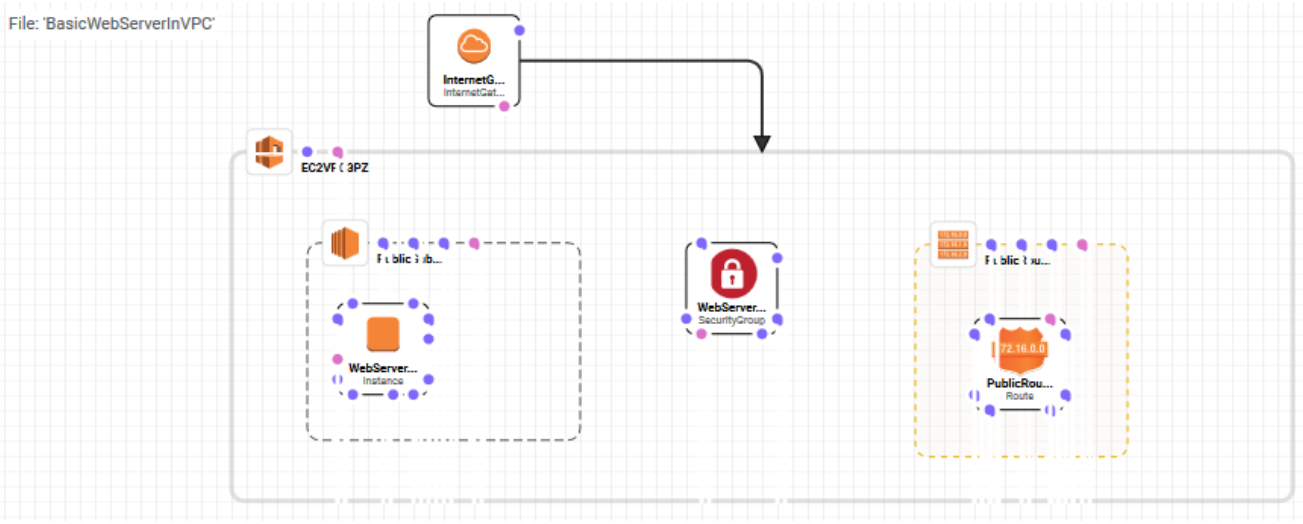


Conectar Recursos.

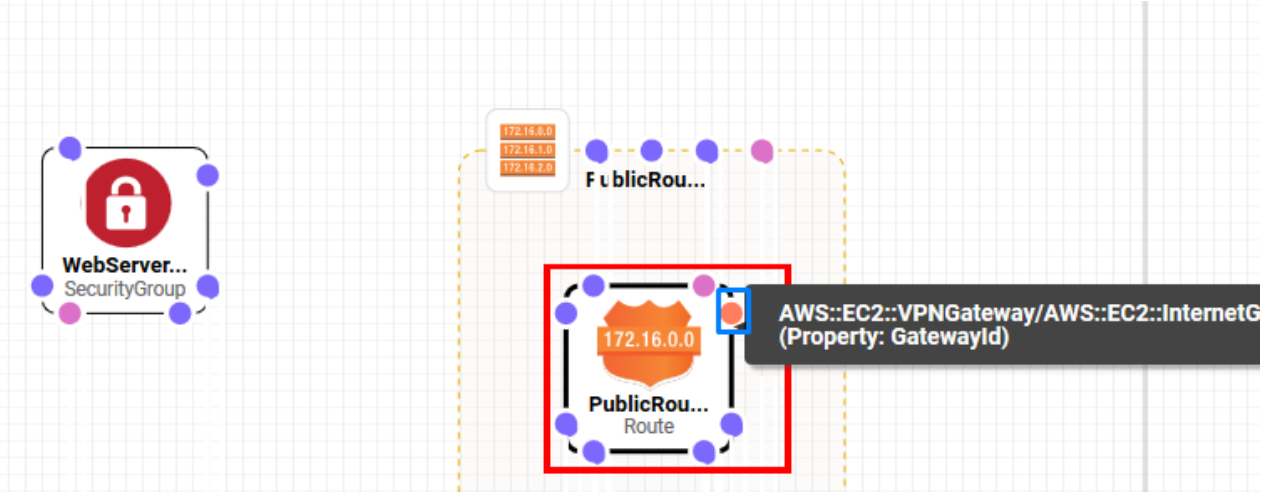
Ahora debemos empezar a gestionar las conexiones entre recursos. Con esto definimos el comportamiento de la VPC y cómo serán tratados los paquetes.

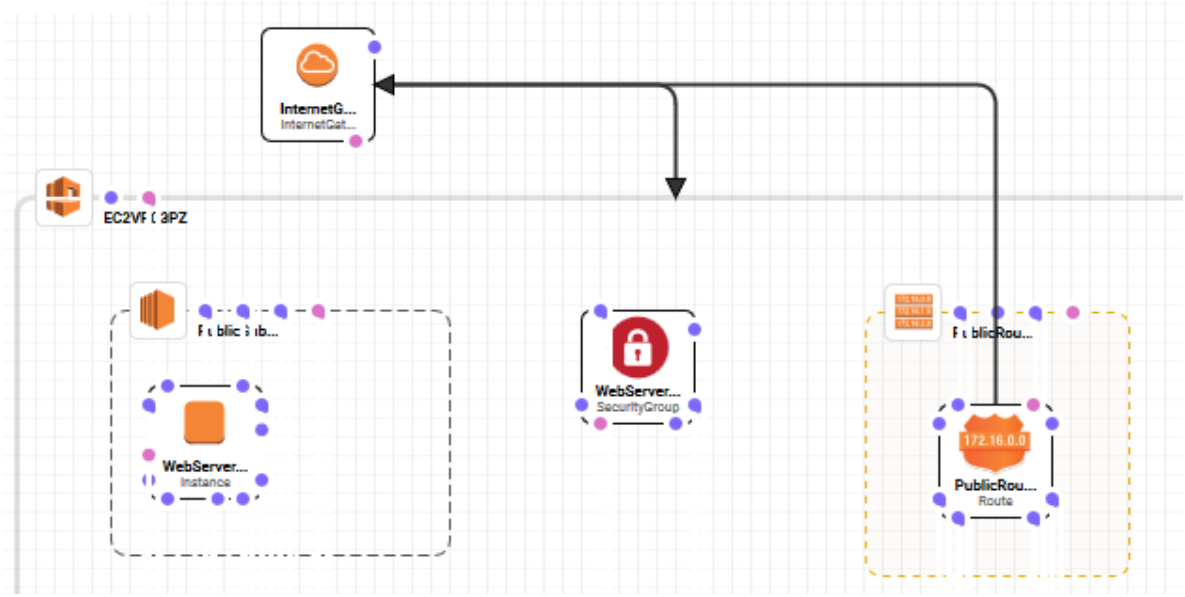
En el recurso **InternetGateway**, coloque el cursor sobre la vinculación de la gateway de Internet **(AWS::EC2::VPCGatewayAttachment**). Arrastre una conexión a la VPC. Cabe resaltar, que el borde de los recursos de destino válidos cambia a color verde. En este caso, la VPC es el único recurso de destino válido.

La conexión debe verse de la siguiente manera:



Para la ruta pública, queremos que el gateway de Internet sea el destino por defecto. Utilice ***GatewayId*** para crear una conexión desde el recurso **`PublicRoute`** al gateway de Internet. Este proceso se aprecia en la siguiente captura de pantalla:

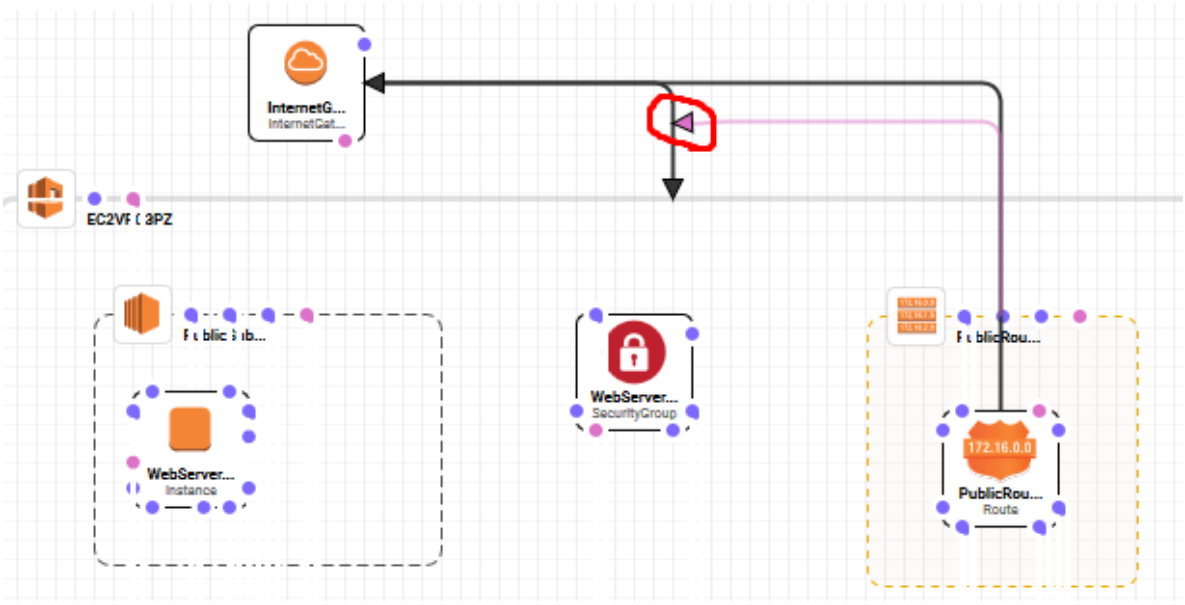




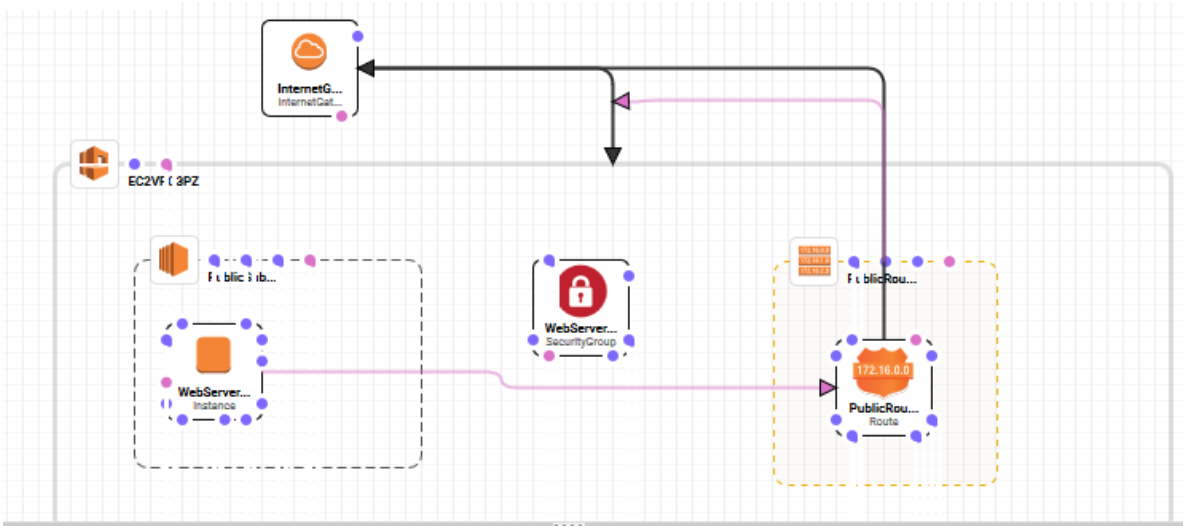
AWS CloudFormation no puede asociar una ruta al gateway de Internet si no se ha realizado la conexión entre el Gateway de Internet y la VPC. Esto significa que tenemos que crear una dependencia explícita en la vinculación gateway de Internet-VPC. Con este proceso, nos aseguramos que nuestros recursos se creen de manera correcta una vez ejecutada nuestra plantilla.

En el recurso **`PublicRoute`**, pase el ratón por encima del punto **DependsOn**. Arrastre una conexión a la vinculación gateway de Internet-VPC (AWS::EC2::VPCGatewayAttachment). Con conexiones DependsOn, AWS CloudFormation Designer crea una dependencia (un atributo DependsOn), donde el recurso de origen depende del recurso de destino.

Al finalizar tendrá el diagrama de la siguiente manera:

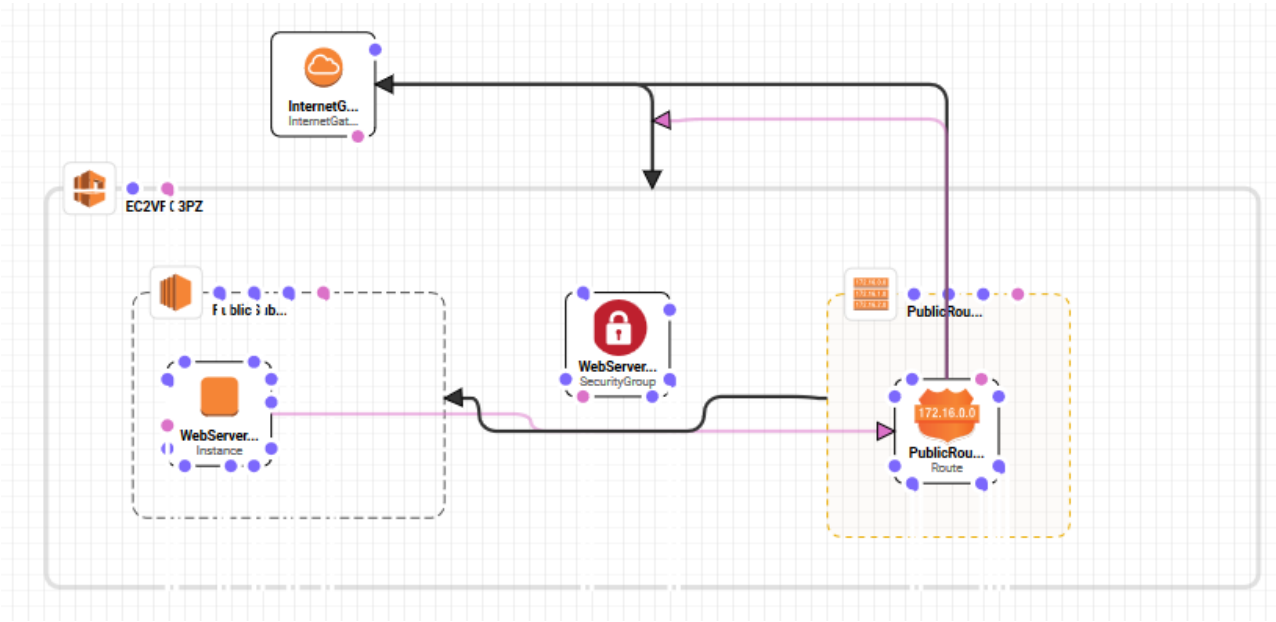


Ahora, cree otra dependencia desde el recurso **`WebServerInstance`** al recurso **`PublicRoute`**. El recurso **`WebServerInstance`** depende de la ruta pública para dirigir el tráfico a través de Internet. Dicha conexión modifica el diagrama así:

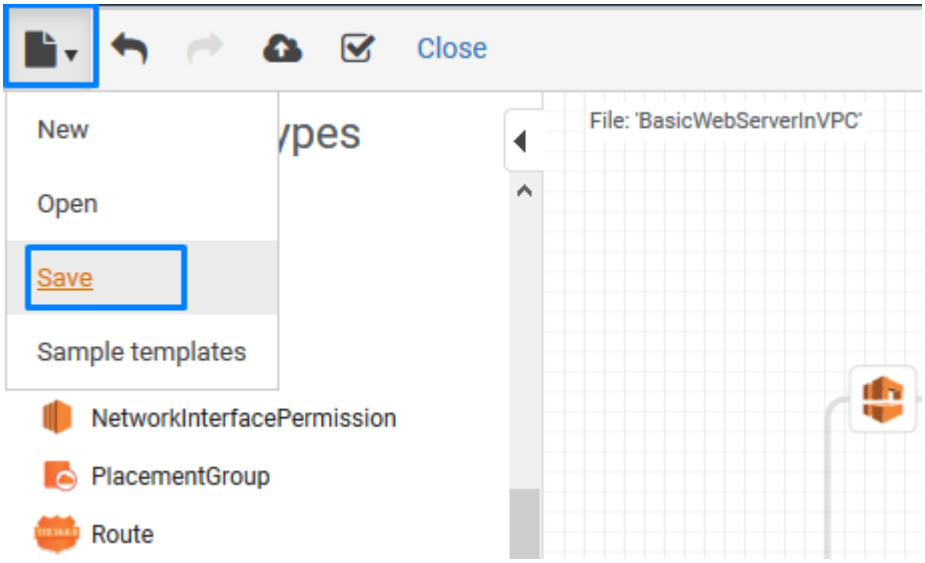


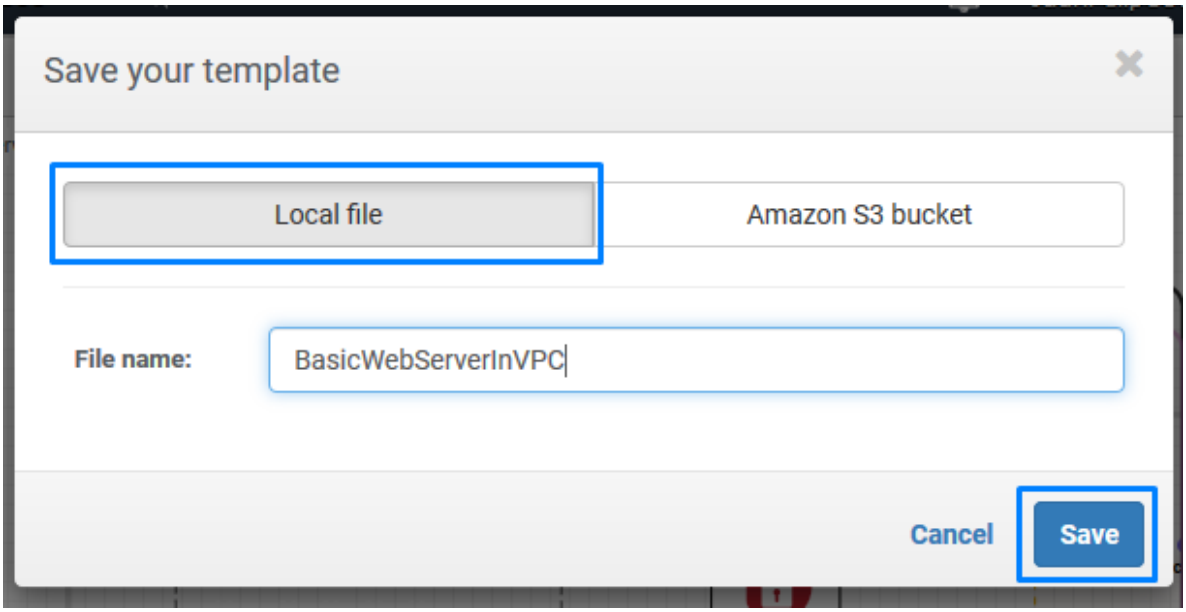
Por último, cree una conexión desde el recurso **`PublicRouteTable`** al recurso **`PublicSubnet`** para asociar la tabla de ruteo y la subred. Ahora la subred pública utilizará la tabla de ruteo público para dirigir el tráfico.

De esta manera, el diagrama final de nuestra infraestructura AWS y sus relaciones será la siguiente:



Puede guardar los avances de la plantilla, haciendo clic en la sección archivo -> save -> Local File. Como se aprecia en las siguientes capturas:





Configuración Parámetros, mapeos y salidas.

Antes de especificar las propiedades de los recursos, tenemos que añadir otros componentes a nuestra plantilla para que resulte más fácil reutilizar la plantilla en futuras ocasiones. En este paso, vamos a utilizar el editor integrado de AWS CloudFormation Designer para añadir parámetros, mapeos y salidas.

Estos parámetros, mapeos y salidas, son valores genéricos de AWS, por lo que siéntase en la libertad de copiar y pegar los fragmentos de código presentados en esta sección.

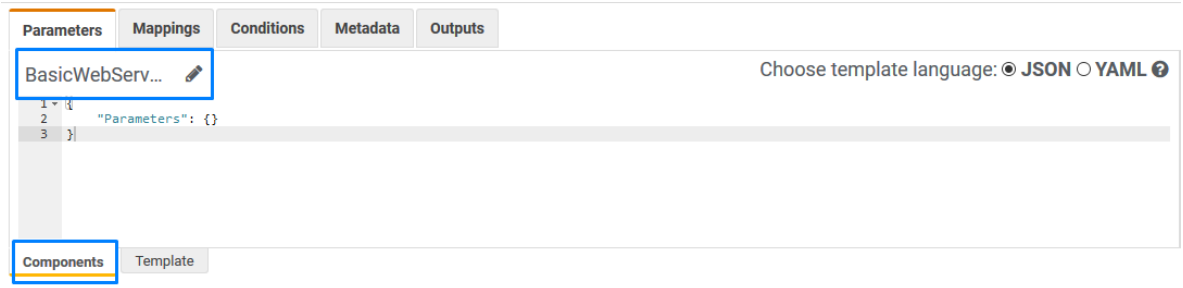
Añadir Parámetros.

Los parámetros son valores de entrada que se especifican al momento de crear una pila (Ejecución de la Plantilla). Son útiles para evitar tener valores codificados de forma rígida en las plantillas. Por ejemplo, no es necesario codificar de manera rígida el tipo de instancia del servidor web en la plantilla; en su lugar, puede utilizar un parámetro para especificar el tipo de instancia (t2.micro, t2.medium, entre otros). De esta forma, puede utilizar la misma plantilla para crear varios servidores web con diferentes tipos de instancias.

En AWS CloudFormation Designer, existen 2 niveles de edición: el nivel de plantilla y el nivel de recursos. En el nivel de la plantilla, puede editar el resto de las secciones de una plantilla, como, por ejemplo, los parámetros, mapeos y salidas de la plantilla, excepto la sección Resources. En el nivel de recursos, puede editar las propiedades de recursos y atributos.

Al hacer clic en una zona abierta en el canvas esto le permite editar los componentes de nivel de plantilla. Para editar los componentes de nivel de recursos, seleccione un recurso.

En el panel de editor integrado, elija la pestaña Parameters (Parámetros) en la vista Components (Componentes).



El siguiente fragmento de código JSON añade parámetros para especificar el tipo de instancia del servidor web, un nombre de par de claves de Amazon EC2 para acceso de SSH al servidor web y el rango de direcciones IP que se puede utilizar para obtener acceso al servidor web mediante SSH.

```
{
  "Parameters": {
    "InstanceType" : {
      "Description" : "WebServer EC2 instance type",
      "Type" : "String",
      "Default" : "t2.micro",
      "AllowedValues" : [
        "t1.micro",
        "t2.nano",
        "t2.micro",
        "t2.small",
        "t2.medium",
        "t2.large",
        "m1.small",
        "m1.medium",
        "m1.large",
        "m1.xlarge",
        "m2.xlarge",
        "m2.2xlarge",
```

```
"m2.4xlarge",
"m3.medium",
"m3.large",
"m3.xlarge",
"m3.2xlarge",
"m4.large",
"m4.xlarge",
"m4.2xlarge",
"m4.4xlarge",
"m4.10xlarge",
"c1.medium",
"c1.xlarge",
"c3.large",
"c3.xlarge",
"c3.2xlarge",
"c3.4xlarge",
"c3.8xlarge",
"c4.large",
"c4.xlarge",
"c4.2xlarge",
"c4.4xlarge",
"c4.8xlarge",
"g2.2xlarge",
"g2.8xlarge",
"r3.large",
"r3.xlarge",
"r3.2xlarge",
"r3.4xlarge",
"r3.8xlarge",
"i2.xlarge",
```

```
        "i2.2xlarge",
        "i2.4xlarge",
        "i2.8xlarge",
        "d2.xlarge",
        "d2.2xlarge",
        "d2.4xlarge",
        "d2.8xlarge",
        "hi1.4xlarge",
        "hs1.8xlarge",
        "cr1.8xlarge",
        "cc2.8xlarge",
        "cg1.4xlarge"
      ],
      "ConstraintDescription" : "must be a valid EC2 instance type."
    },
    "KeyName": {
      "Description": "Name of an EC2 KeyPair to enable SSH access to
the instance.",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription": "must be the name of an existing EC2
KeyPair."
    },
    "SSHLocation": {
      "Description": " The IP address range that can be used to
access the web server using SSH.",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern":
"(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
```

```
      "ConstraintDescription": "must be a valid IP CIDR range of the
form x.x.x.x/x."

    }

  }

}
```

Después de copiar los parámetros, la plantilla deberá verse de la siguiente manera:

Añadir Mapeos.

Los mapeos son un conjunto de claves que se asocian a un conjunto de pares de nombre-valor. Son útiles para especificar valores basados en un valor de parámetro de entrada. En este tutorial, utilizaremos un mapeo para especificar un ID de AMI para una instancia EC2 en función del tipo de instancia y la región en la que se crea la pila.

En el panel del editor integrado, elija la pestaña Mappings (Mapeos). Copie los siguientes mapeos de JSON y péguelos en editor integrado.

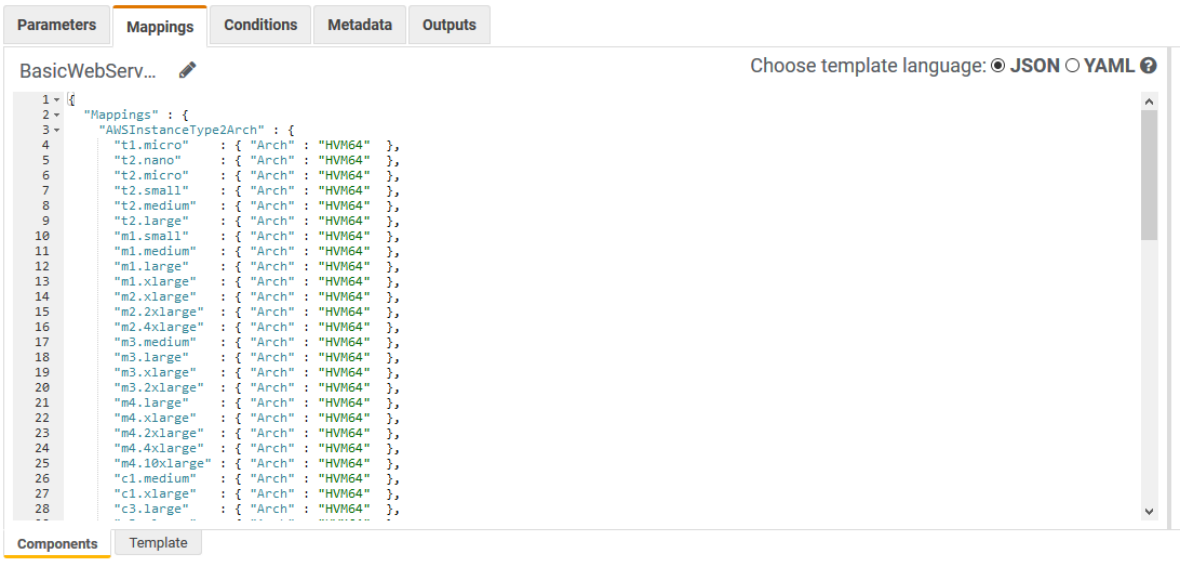
```
{

  "Mappings" : {

    "AWSInstanceType2Arch" : {

      "t1.micro"    : { "Arch" : "HVM64"  },

      "t2.nano"     : { "Arch" : "HVM64"  },
```

```
    "t2.micro"    : { "Arch" : "HVM64"  },
    "t2.small"    : { "Arch" : "HVM64"  },
    "t2.medium"   : { "Arch" : "HVM64"  },
    "t2.large"    : { "Arch" : "HVM64"  },
    "m1.small"    : { "Arch" : "HVM64"  },
    "m1.medium"   : { "Arch" : "HVM64"  },
    "m1.large"    : { "Arch" : "HVM64"  },
    "m1.xlarge"   : { "Arch" : "HVM64"  },
    "m2.xlarge"   : { "Arch" : "HVM64"  },
    "m2.2xlarge"  : { "Arch" : "HVM64"  },
    "m2.4xlarge"  : { "Arch" : "HVM64"  },
    "m3.medium"   : { "Arch" : "HVM64"  },
    "m3.large"    : { "Arch" : "HVM64"  },
    "m3.xlarge"   : { "Arch" : "HVM64"  },
    "m3.2xlarge"  : { "Arch" : "HVM64"  },
    "m4.large"    : { "Arch" : "HVM64"  },
    "m4.xlarge"   : { "Arch" : "HVM64"  },
    "m4.2xlarge"  : { "Arch" : "HVM64"  },
    "m4.4xlarge"  : { "Arch" : "HVM64"  },
    "m4.10xlarge" : { "Arch" : "HVM64"  },
    "c1.medium"   : { "Arch" : "HVM64"  },
    "c1.xlarge"   : { "Arch" : "HVM64"  },
    "c3.large"    : { "Arch" : "HVM64"  },
    "c3.xlarge"   : { "Arch" : "HVM64"  },
    "c3.2xlarge"  : { "Arch" : "HVM64"  },
    "c3.4xlarge"  : { "Arch" : "HVM64"  },
    "c3.8xlarge"  : { "Arch" : "HVM64"  },
    "c4.large"    : { "Arch" : "HVM64"  },
    "c4.xlarge"   : { "Arch" : "HVM64"  },
    "c4.2xlarge"  : { "Arch" : "HVM64"  },
```

```
      "c4.4xlarge"  : { "Arch" : "HVM64"  },
      "c4.8xlarge"  : { "Arch" : "HVM64"  },
      "g2.2xlarge"  : { "Arch" : "HVMG2"  },
      "g2.8xlarge"  : { "Arch" : "HVMG2"  },
      "r3.large"    : { "Arch" : "HVM64"  },
      "r3.xlarge"   : { "Arch" : "HVM64"  },
      "r3.2xlarge"  : { "Arch" : "HVM64"  },
      "r3.4xlarge"  : { "Arch" : "HVM64"  },
      "r3.8xlarge"  : { "Arch" : "HVM64"  },
      "i2.xlarge"   : { "Arch" : "HVM64"  },
      "i2.2xlarge"  : { "Arch" : "HVM64"  },
      "i2.4xlarge"  : { "Arch" : "HVM64"  },
      "i2.8xlarge"  : { "Arch" : "HVM64"  },
      "d2.xlarge"   : { "Arch" : "HVM64"  },
      "d2.2xlarge"  : { "Arch" : "HVM64"  },
      "d2.4xlarge"  : { "Arch" : "HVM64"  },
      "d2.8xlarge"  : { "Arch" : "HVM64"  },
      "hi1.4xlarge" : { "Arch" : "HVM64"  },
      "hs1.8xlarge" : { "Arch" : "HVM64"  },
      "cr1.8xlarge" : { "Arch" : "HVM64"  },
      "cc2.8xlarge" : { "Arch" : "HVM64"  }
    },
    "AWSRegionArch2AMI" : {
      "us-east-1"        : {"HVM64" : "ami-0ff8a91507f77f867",
"HVMG2" : "ami-0a584ac55a7631c0c"},
      "us-west-2"        : {"HVM64" : "ami-a0cfeed8", "HVMG2" :
"ami-0e09505bc235aa82d"},
      "us-west-1"        : {"HVM64" : "ami-0bdb828fd58c52235",
"HVMG2" : "ami-066ee5fd4a9ef77f1"},
      "eu-west-1"        : {"HVM64" : "ami-047bb4163c506cd98",
"HVMG2" : "ami-0a7c483d527806435"},
```

```
      "eu-west-2"        : {"HVM64" : "ami-f976839e", "HVMG2" :
"NOT_SUPPORTED"},

      "eu-west-3"        : {"HVM64" : "ami-0ebc281c20e89ba4b",
"HVMG2" : "NOT_SUPPORTED"},

      "eu-central-1"     : {"HVM64" : "ami-0233214e13e500f77",
"HVMG2" : "ami-06223d46a6d0661c7"},

      "ap-northeast-1"   : {"HVM64" : "ami-06cd52961ce9f0d85",
"HVMG2" : "ami-053cdd503598e4a9d"},

      "ap-northeast-2"   : {"HVM64" : "ami-0a10b2721688ce9d2",
"HVMG2" : "NOT_SUPPORTED"},

      "ap-northeast-3"   : {"HVM64" : "ami-0d98120a9fb693f07",
"HVMG2" : "NOT_SUPPORTED"},

      "ap-southeast-1"   : {"HVM64" : "ami-08569b978cc4dfa10",
"HVMG2" : "ami-0be9df32ae9f92309"},

      "ap-southeast-2"   : {"HVM64" : "ami-09b42976632b27e9b",
"HVMG2" : "ami-0a9ce9fecc3d1daf8"},

      "ap-south-1"       : {"HVM64" : "ami-0912f71e06545ad88",
"HVMG2" : "ami-097b15e89dbdcfcf4"},

      "us-east-2"        : {"HVM64" : "ami-0b59bfac6be064b78",
"HVMG2" : "NOT_SUPPORTED"},

      "ca-central-1"     : {"HVM64" : "ami-0b18956f", "HVMG2" :
"NOT_SUPPORTED"},

      "sa-east-1"        : {"HVM64" : "ami-07b14488da8ea02a0",
"HVMG2" : "NOT_SUPPORTED"},

      "cn-north-1"       : {"HVM64" : "ami-0a4eaf6c4454eda75",
"HVMG2" : "NOT_SUPPORTED"},

      "cn-northwest-1"   : {"HVM64" : "ami-6b6a7d09", "HVMG2" :
"NOT_SUPPORTED"}

    }

  }

}
```

Al finalizar su plantilla debe ser similar a la siguiente captura de pantalla:



Añadir Salidas

Las salidas declaran valores que desea que estén disponibles para una llamada a la API de stacks o a través de la pestaña Outputs (Salidas) del AWS CloudFormation una vez ejecutada la plantilla. En este laboratorio, se genera una salida con la URL del sitio web para que pueda ver fácilmente el sitio web una vez se haya creado.

En el panel del editor integrado, seleccione la pestaña Outputs (Salidas). Copie la siguiente salida JSON y péguela en editor integrado.

```
{
  "Outputs": {
    "URL": {
      "Value": {
        "Fn::Join": [
          "",
          [
            "http://",
            {
              "Fn::GetAtt": [
                "WebServerInstance",
                "PublicIp"
```

```
                ]
              }
            ]
          ]
        },
        "Description": "Newly created application URL"
      }
    }
}
```

Al finalizar, debería tener la configuración de salidas (outputs) como se aprecia en la siguiente captura:

RECUERDE GUARDAR LOS CAMBIOS!!!

Especificar Propiedades de los recursos.

Muchos recursos necesitan especificar sus propiedades para definir sus configuraciones o ajustes, como, por ejemplo, el tipo de instancia que se debe utilizar para el servidor web. De forma similar a lo que hicimos en la sección anterior, vamos a utilizar el editor integrado de AWS CloudFormation Designer para especificar propiedades de los recursos.

Estas configuraciones, serán dadas por la guía para agilizar el proceso de ejecución de la plantilla. Siéntase en la libertad de copiar y pegar los fragmentos de código presentados en esta sección.

Propiedades VPC.

En el canvas de AWS CloudFormation Designer, elija el recurso **VPC**. El editor integrado muestra los componentes de nivel de recurso que puede editar, como, por ejemplo, las propiedades de recursos y atributos. En el panel del editor integrado, elija la pestaña Properties (Propiedades). Copie el siguiente fragmento de código JSON y péguelo en el editor integrado. Este fragmento de código especifica los ajustes de DNS y el bloque de CIDR de la VPC.

```
"Properties": {

                    "EnableDnsSupport": "true",
                    "EnableDnsHostnames": "true",
                    "CidrBlock": "10.0.0.0/16"
                }
```

Las propiedades de la VPC deben quedar así:

Propiedades Subnet.

Agregue la siguiente propiedad del bloque de CIDR después de la propiedad del ID de VPC. AWS CloudFormation Designer añadió automáticamente la propiedad del ID de VPC cuando arrastró la subred dentro de la VPC.

```
{
    "Resources": {
        "PublicSubnet": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "CidrBlock": "10.0.0.0/24"
            }
        }
    }
}
```

Las propiedades de la subnet quedan cómo se aprecian a continuación:

Propiedades PublicRoute

Agregue la siguiente propiedad de bloque de CIDR de destino, que dirige todo el tráfico a gateway de Internet en el recurso **PublicRoute**

```
{
    "Resources": {
        "PublicRoute": {
            "Type": "AWS::EC2::Route",
            "Properties": {
                "DestinationCidrBlock": "0.0.0.0/0",
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "GatewayId": {
                    "Ref": "InternetGateway"
                }
            }
        }
    }
}
```

Debe observar las propiedades de la siguiente manera.

Propiedades Security Group.

Añada las siguientes reglas entrantes que determinan el tráfico que puede llegar a la instancia del servidor web. Las reglas permiten todo el tráfico HTTP y determinado tráfico SSH.

```
{
    "Resources": {
        "WebServerSecurityGroup": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "GroupDescription": "Allow access from HTTP and SSH
traffic",
                "SecurityGroupIngress": [
                    {
                        "IpProtocol": "tcp",
                        "FromPort": "80",
```

```
                    "ToPort": "80",
                    "CidrIp": "0.0.0.0/0"
                },
                {
                    "IpProtocol": "tcp",
                    "FromPort": "22",
                    "ToPort": "22",
                    "CidrIp": {
                        "Ref": "SSHLocation"
                    }
                }
            ]
        }
    }
}
```

Propiedades Instancia WebServerInstance.

Ahora, se debe especificar un número de propiedades para la instancia del servidor web, por lo que vamos a configurar tan solo algunas propiedades para realizar este laboratorio. Las propiedades InstanceType e ImageId utilizan los valores de los parámetros y mapeos que especificamos en la sección anterior.

La propiedad NetworkInterfaces especifica los ajustes de red para la instancia del servidor web. Esta propiedad nos permite asociar el grupo de seguridad y la subred a la instancia. Aunque AWS CloudFormation Designer utilizó la propiedad SubnetId para asociar la instancia a la subred, tenemos que utilizar la propiedad NetworkInterfaces porque es la única forma de dar al servidor web una dirección IP pública. Cuando especifica la propiedad NetworkInterfaces, debe especificar la subred y el grupo de seguridad dentro de dicha propiedad.

En la propiedad UserData, se especifican los scripts de configuración que se ejecutan una vez que la instancia está operativa. Toda la información de configuración se define en los metadatos de la instancia, que añadiremos en el siguiente paso.

Sustituya todas las propiedades por el siguiente fragmento de código:

```
{
    "Resources": {
        "WebServerInstance": {
            "Type": "AWS::EC2::Instance",
            "Properties": {

                "InstanceType": {
                  "Ref": "InstanceType"
                },
                "ImageId": {
                  "Fn::FindInMap": [
                    "AWSRegionArch2AMI",
                    {
                      "Ref": "AWS::Region"
                    },
                    {
                      "Fn::FindInMap": [
                        "AWSInstanceType2Arch",
                        {
                          "Ref": "InstanceType"
                        },
                        "Arch"
                      ]
                    }
                  ]
                },
                "KeyName": {
                  "Ref": "KeyName"
                },
```

```
                    "NetworkInterfaces": [
                       {
                        "GroupSet": [
                          {
                            "Ref": "WebServerSecurityGroup"
                          }
                        ],
                        "AssociatePublicIpAddress": "true",
                        "DeviceIndex": "0",
                        "DeleteOnTermination": "true",
                        "SubnetId": {
                          "Ref": "PublicSubnet"
                        }
                      }
                    ],

                    "UserData": {
                      "Fn::Base64": {
                        "Fn::Join": [
                          "",
                          [
                            "#!/bin/bash -xe\n",
                            "yum install -y aws-cfn-bootstrap\n",
                            "# Install the files and packages from the
metadata\n",
                            "/opt/aws/bin/cfn-init -v ",
                            "         --stack ",
                            {
                              "Ref": "AWS::StackName"
                            },
```

```
                        "         --resource WebServerInstance ",
                        "         --configsets All ",
                        "         --region ",
                        {
                          "Ref": "AWS::Region"
                        },
                        "\n",
                        "# Signal the status from cfn-init\n",
                        "/opt/aws/bin/cfn-signal -e $? ",
                        "         --stack ",
                        {
                          "Ref": "AWS::StackName"
                        },
                        "         --resource WebServerInstance ",
                        "         --region ",
                        {
                          "Ref": "AWS::Region"
                        },
                        "\n"
                      ]
                    ]
                  }
                }
              }
            }
          }
        }
```

De esta manera, las propiedades de la instancia se aprecian a continuación:

Guía Elaborada por: Juan Felipe Gómez Manzanares.
Ingeniero Telemático

Metadatos Instancia WebServerInstance.

Elija el recurso **`WebServerInstance`** y, a continuación, elija la pestaña Metadata (Metadatos) en el panel del editor integrado. Los Metadatos deben quedar de la siguiente manera.

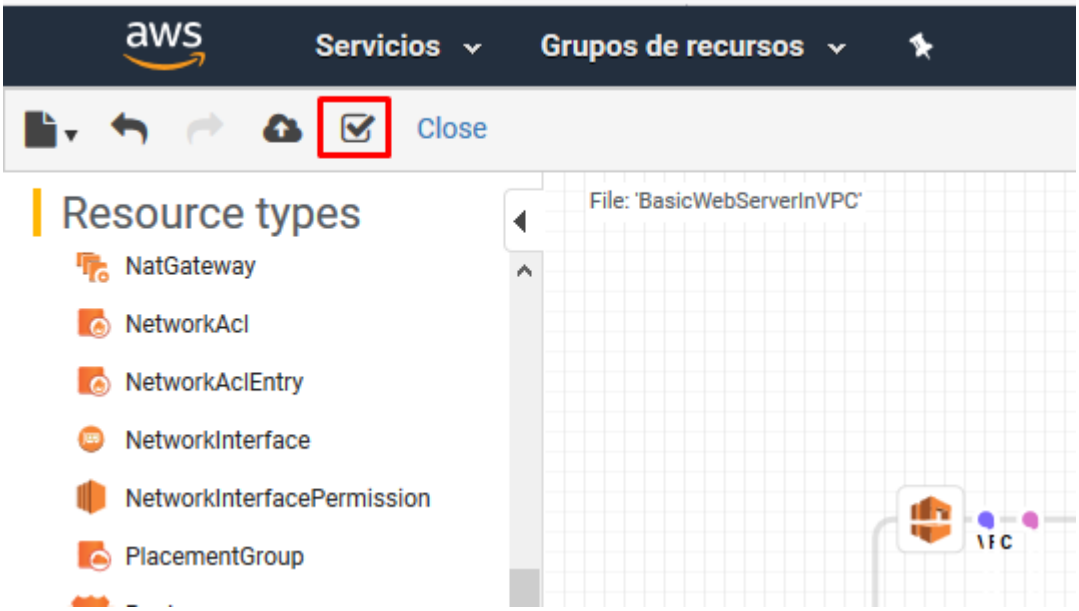
```
{
    "Resources": {
        "WebServerInstance": {
            "Type": "AWS::EC2::Instance",
            "Metadata": {
                "AWS::CloudFormation::Designer": {
                    "id": "f05d419d-9129-4547-9eba-bfd38fe74f2e"
                },
                    "AWS::CloudFormation::Init" : {
                  "configSets" : {
```

```
                    "All" : [ "ConfigureSampleApp" ]
                  },
                  "ConfigureSampleApp" : {
                    "packages" : {
                      "yum" : {
                        "httpd" : []
                      }
                    },
                    "files" : {
                      "/var/www/html/index.html" : {
                        "content" : { "Fn::Join" : ["\n", [
                          "<h1>Felicitaciones. Acaba de implementar
su primera plantilla en CloudFomration!!!</h1>"
                        ]]},
                        "mode"    : "000644",
                        "owner"   : "root",
                        "group"   : "root"
                      }
                    },
                    "services" : {
                      "sysvinit" : {
                        "httpd"        :  {  "enabled"  :  "true",
"ensureRunning" : "true" }
                      }
                    }
                  }
                }
              }
          }
      }
```

}

Validar Plantilla.

Ahora ya tendríamos lista nuestra plantilla para ejecución, sólo nos hace falta un paso. El último paso es validar que nuestra plantilla se encuentre correcta, utilizando el botón validar template cómo se aprecia en la siguiente captura.



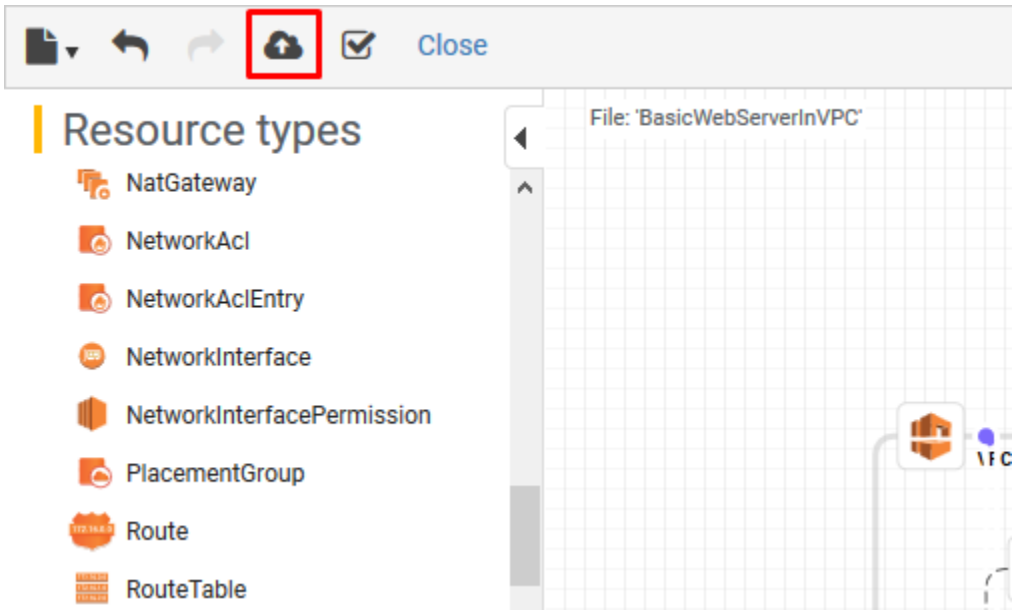
Si hemos seguido cada uno de los pasos de esta guía tendremos el siguiente mensaje:

RECUERDE GUARDAR LA PLANTILLA!!!!

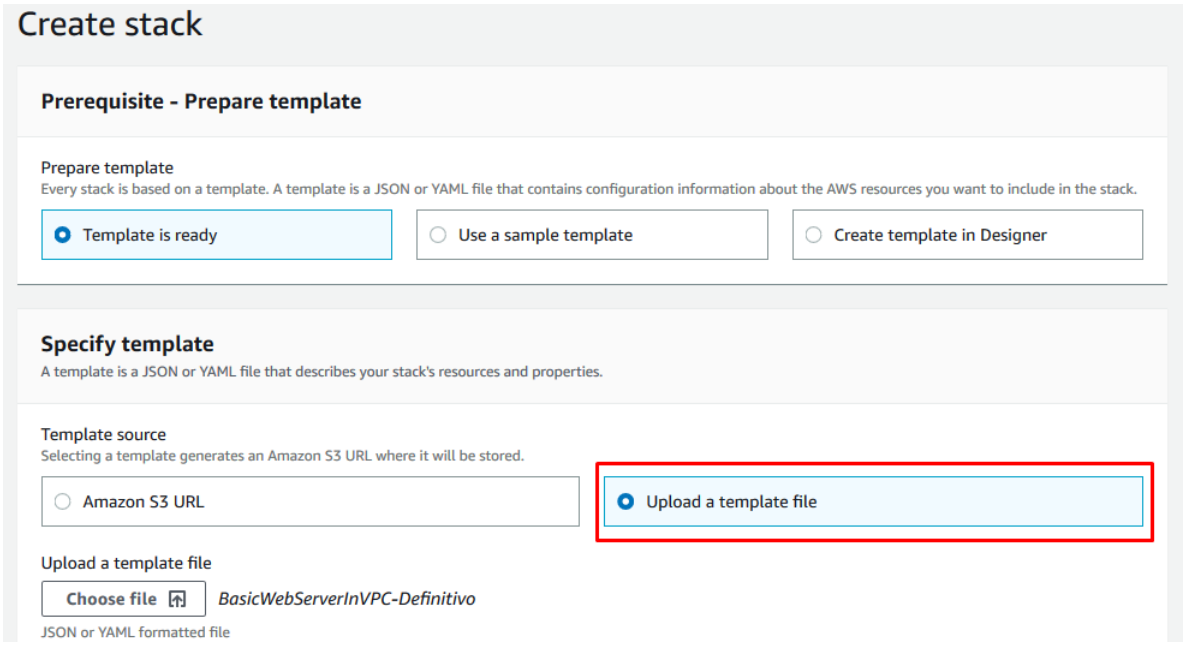
Guía Elaborada por: Juan Felipe Gómez Manzanares.
Ingeniero Telemático

Crear Stack y Ejecutar Plantilla.

Una vez terminada la creación de la plantilla, ya podremos realizar la ejecución de la misma para crear nuestra infraestructura AWS de manera automática a través del AWS CloudFormation. Para iniciar el proceso de ejecución, damos clic en la opción crear stack:



Nos aparecerá el siguiente menú, dónde seleccionaremos utilizar un template y subir un template file. Aquí subiremos la plantilla que acabamos de desarrollar:

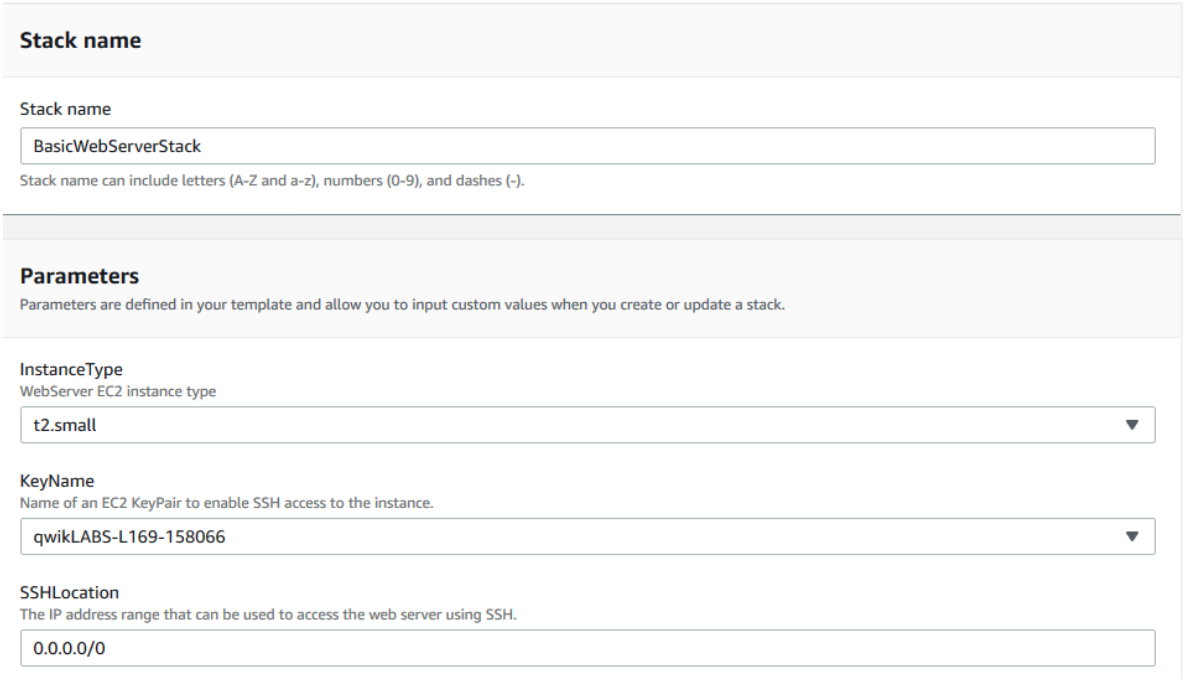


Guía Elaborada por: Juan Felipe Gómez Manzanares.
Ingeniero Telemático

Algo importante que debe considerarse, es que Amazon creará un nuevo Bucket S3 para guardar el template, así, puede utilizarlo en un futuro de manera muy rápida.

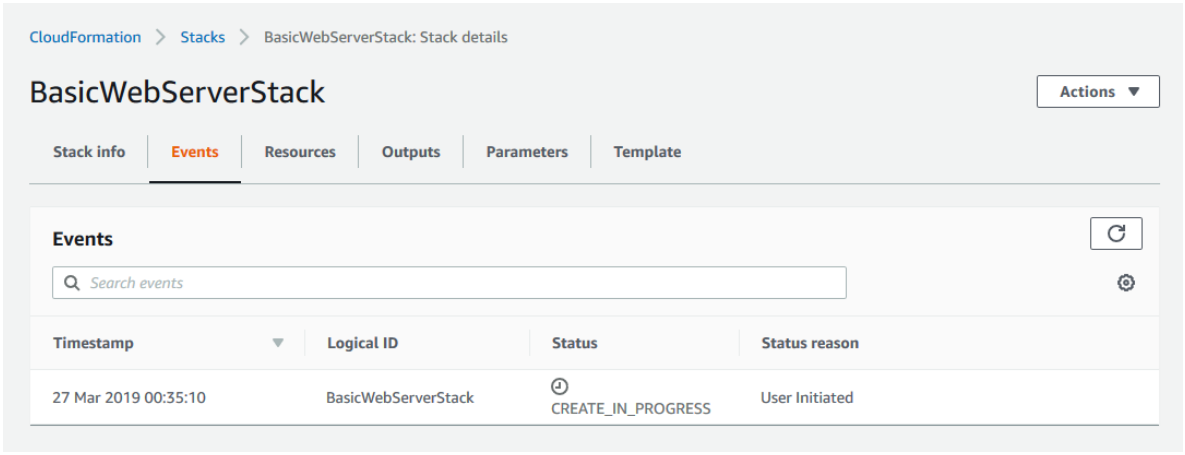
En la sección Specify Details (Especificar detalles), introduzca un nombre de pila en el campo Stack name (Nombre de pila). Para este ejemplo, use **`BasicWebServerStack`**.

Los parámetros, que vemos en la creación del stack, son los que definimos previamente en la plantilla. Seleccione el tipo de instancia que está dispuesto a pagar, el nombre de las llaves a utilizar y la configuración de las reglas SSH. En esté caso, maneje la configuración cómo aparece a continuación:



Damos clic en Next, y nos aparece la ventana para configurar las opciones de nuestro stack. Podemos configurar las etiquetas, roles IAM, políticas, configuraciones de rollback y demás opciones que podemos dejar en blanco en este momento.

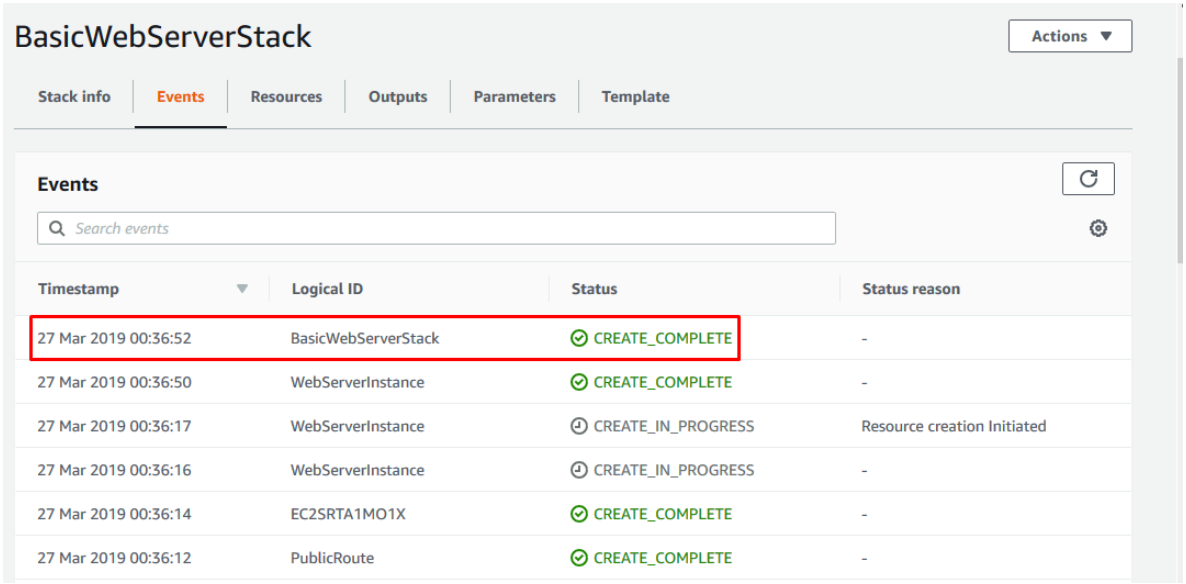
Por último, validamos toda la información que nos muestra AWS CloudFormation y damos click en Create Stack. Nos aparece el siguiente mensaje:}

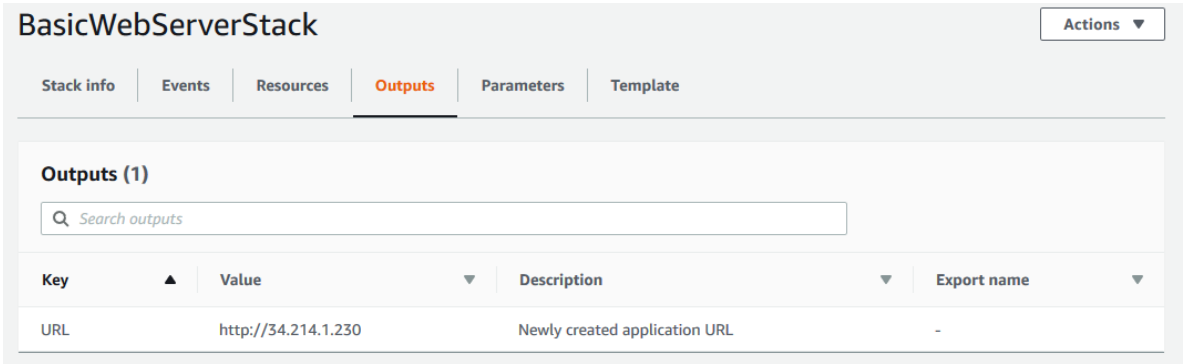


Por ahora sólo debemos esperar que nuestro template se ejecute y al final podremos ver que nuestra infraestructura de AWS se encuentra operativa.

Validación y Pruebas.

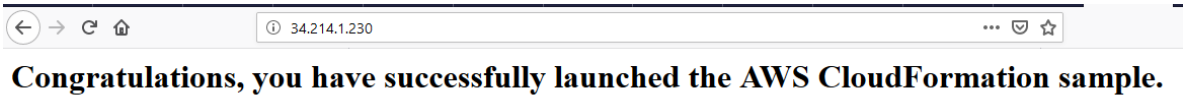
Si todo sale bien, obtendremos el siguiente mensaje:





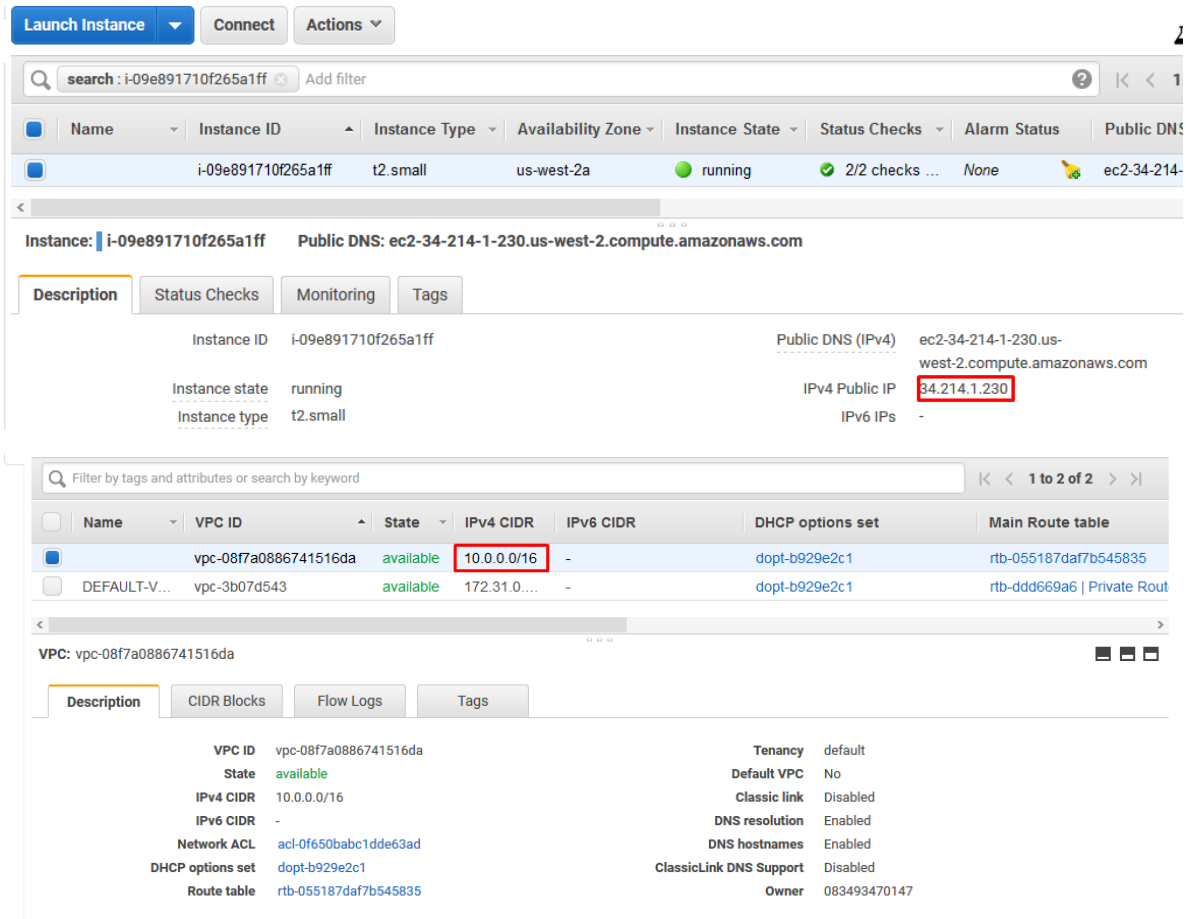
Guía Elaborada por: Juan Felipe Gómez Manzanares.
Ingeniero Telemático

Ingresando a la dirección IP que nos muestra desde el Output, obtendremos:

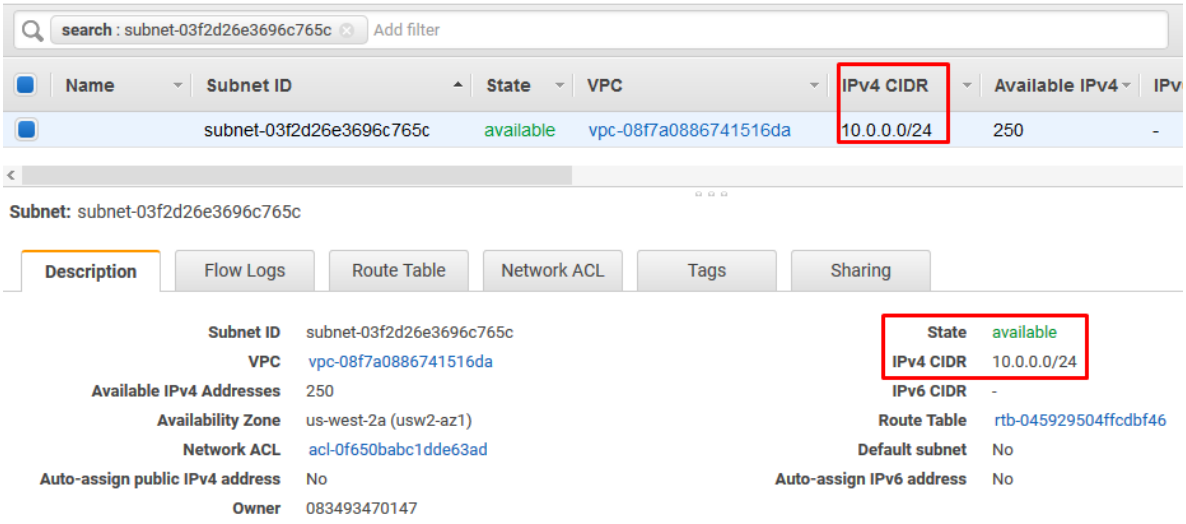


Demostrando que nuestro Templete se ejecuta de manera correcta.

Para finalizar Validamos que todos nuestros recursos se crean de manera correcta.



Guía Elaborada por: Juan Felipe Gómez Manzanares.
Ingeniero Telemático



Bibliografía

Amazon Web Services. (2019). *Tutorial: Procedimiento de creación de un servidor web básico con AWS CloudFormation Designer.* Obtenido de https://docs.aws.amazon.com/es_es/AWSCloudFormation/latest/UserGuide/working-with-templates-cfn-designer-walkthrough-createbasicwebserver.html