



AWS

CLOUDFORMATION

Herramienta de aprovisionamiento

DESCRIPCIÓN

Un proyecto dedicado a la investigación y uso de la herramienta de aprovisionamiento (IaC) de AWS, CloudFormation

Javier Crespillo

2º ASIR IES GONZALO NAZARENO

AWS	CLOUDFORMATION	0
1.	Introducción al proyecto.	2
1.1	¿Por qué éste proyecto?	2
1.2	¿Qué intento conseguir con este proyecto?	2
1.3	¿Qué he conseguido durante el proyecto?	3
2.	Requisitos para utilizar la herramienta.	3
3.	Conocimientos previos.	4
4.	Funcionamiento de AWS Cloud Formation	6
4.1	Desglose de una plantilla	8
4.1.1	Formato	8
4.1.2	Description	8
4.1.3	Metadata	8
4.1.4	Parameters	10
4.1.5	Reglas	10
4.1.6	Mapping	11
4.1.7	Condiciones	12
4.1.8	Transforms	12
4.1.9	Resources	12
4.1.10	Outputs	13
4.1.11	Pequeño resumen:	13
5.	Creación de un escenario simple de prueba	13
6.	Diferencias entre CloudFormation y otros IaC	26
7.	Conclusiones, propuestas y dificultades	28
8.	Referencias	28

1. Introducción al proyecto.

1.1 ¿Por qué éste proyecto?

En el mercado tecnológico actual, el sector que indudablemente ha crecido más los últimos años es el del Cloud Computing. Tanto ha sido así, que las empresas más grandes del mundo han sacado sus propias versiones del cloud, arrancando esta tecnología como un modelo de negocio a nivel mundial con un gran soporte en la totalidad del globo y en una gran cantidad de idiomas.

Empresas de todo el mundo utilizan a diario tecnologías basadas en la nube para realizar sus operaciones. ¿Por qué? Porque es mucho más barato que desplegar tu propia infraestructura de servidores, y además eliminas el factor de mantenimiento que todo eso conlleva.

Por lo tanto, en ésta practica he querido concentrarme en uno de los gigantes del cloud que existen actualmente, como es el caso de Amazon AWS.

A lo largo de mi aprendizaje en el curso de ASIR, hemos visto y tocado tecnologías que podrían ser usadas para configurar estos servicios en el cloud, también conocida como IaC (Infrastructure as code), por lo que me ha parecido relevante, tomar una de las opciones que nos facilita AmazonAWS para la configuración y aprovisionamiento automático de sus máquinas, dado que creo que esta herramienta es una apuesta segura como tecnología que necesitaré a lo largo de mi carrera profesional.

1.2 ¿Qué intento conseguir con este proyecto?

Este proyecto nació como una idea de aprender a manejar una herramienta de IaC similar a las que ya había tocado previamente. Durante el curso de ASIR aprendimos a manejar la herramienta de aprovisionamiento ANSIBLE, por lo que durante ésta practica he intentado conocer las diferencias y similitudes entre ambas, e incluso ser capaz de compararlas con otras a su vez similares como Terraform.

El proyecto, va a ser una investigación de la herramienta, realizando algún despliegue sencillo, pero sin demasiada complejidad, dado que la capa gratuita de amazon no da para más.

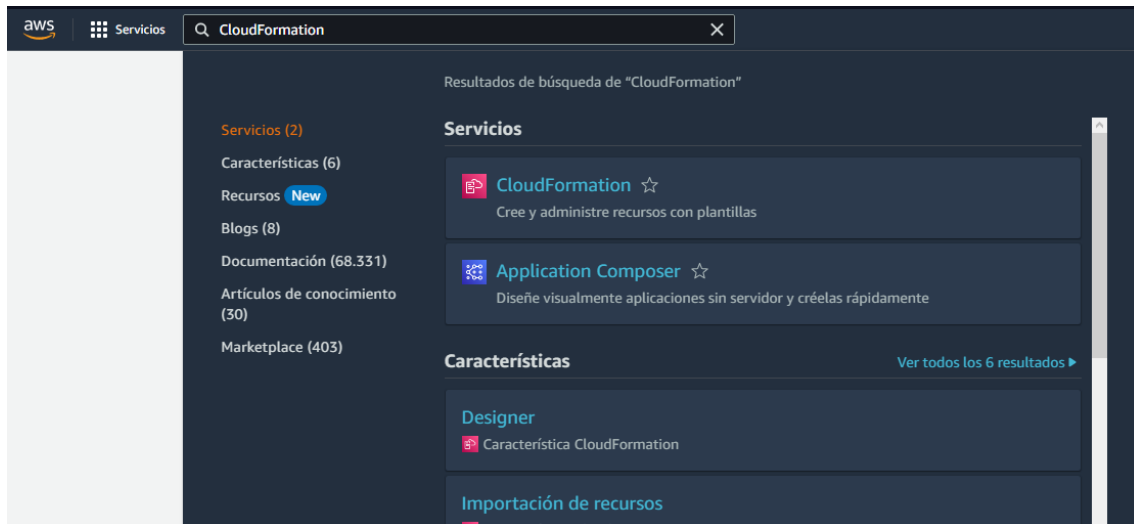
1.3 ¿Qué he conseguido durante el proyecto?

En mi caso, he conseguido aprender (Al menos a un nivel de comprensión medio) el uso y configuración de AWS CloudFormation, el uso de templates, las IAMs necesarias y todo tipo de recursos que nos ofrece la herramienta para el correcto manejo y despliegue de las máquinas.

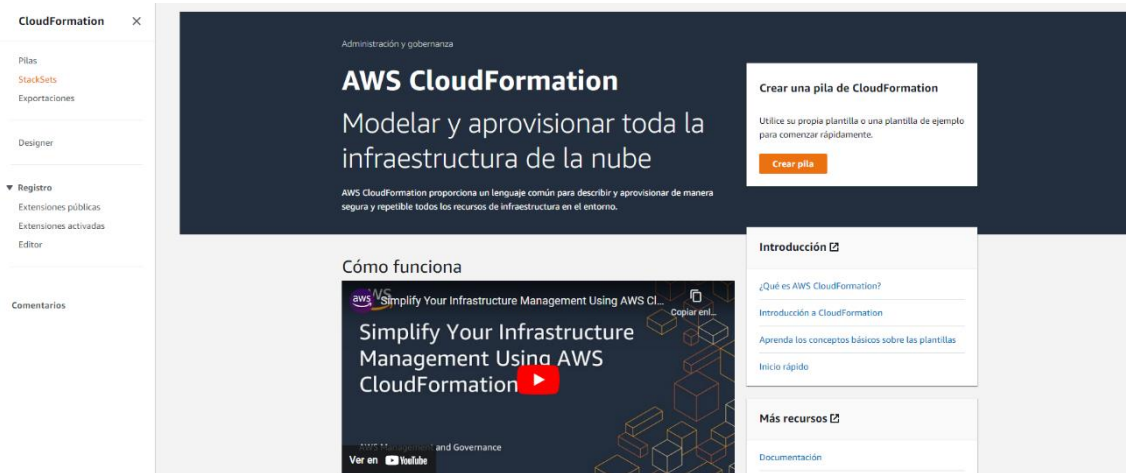
2. Requisitos para utilizar la herramienta.

Para poder practicar con ella vamos a necesitar una cuenta en amazon aws. En mi caso voy a utilizar la capa gratuita pero esta herramienta funciona en todos los planes de amazon aws.

Una vez hecha nuestra cuenta (Nos pedirá registro con tarjeta de crédito), podemos buscar la herramienta que necesitamos en la barra de búsqueda (En nuestro caso CloudFormation)



Una vez dentro se nos desplegarán todo tipo de documentaciones sobre la herramienta, así como los distintos usos de esta.



Y con esto ya podemos empezar a utilizar AWS CloudFormation.

3. Conocimientos previos.

Antes de pasar a la explicación de la herramienta y a la realización y explicación de un despliegue sencillo, creo necesario explicar una serie de conceptos que son relevantes conocer para el correcto entendimiento de este proyecto, así que procedo a ello:

Infraestructura como código (IaC): La infraestructura como código (IaC del inglés Infrastructure as Code) permite gestionar y preparar la infraestructura a través del código, en lugar de hacerlo mediante procesos manuales.

Con este tipo de infraestructura, se crean archivos de configuración que contienen las especificaciones que esta necesita, lo cual facilita la edición y la distribución de las configuraciones. Asimismo, garantiza que siempre se prepare el mismo entorno. La infraestructura como código codifica y documenta las especificaciones para facilitar la gestión de la configuración, y ayuda a evitar los cambios no documentados.

Pilas: Una pila es una colección de recursos de AWS que se puede administrar como una sola unidad. Todos los recursos de una pila se definen mediante la plantilla de AWS CloudFormation de la pila. (También pueden llamarse Stacks)

StackSets: StackSets permite crear, actualizar o eliminar pilas en diferentes cuentas y regiones con una sola operación

IAM: AWS Identity and Access Management (IAM) es un servicio web que le ayuda a controlar de forma segura el acceso a los recursos de AWS. Puedes usar IAM para controlar quién está autenticado (ha iniciado sesión) y autorizado (tiene permisos) para utilizar recursos.



Templates (O plantillas): Se tratan de ficheros JSON o YAML que usa CloudFormation como base para la creación de los recursos AWS. Es el objeto principal de la herramienta y con el que podemos formar y desplegar nuestros sistemas.

4. Funcionamiento de AWS Cloud Formation

Voy a explicar, paso a paso cual sería un proceso de despliegue básico en AWS CloudFormation, haciendo hincapié en los procesos más importantes.

AWS CloudFormation comienza en la plantilla yaml/json que queramos crear, puesto que será esto lo que le de forma a nuestro despliegue. Las plantillas se componen de una serie de parámetros, en su mayoría opcionales, que describen de forma detallada todos los procesos que va a realizar la herramienta para el despliegue de nuestras maquinas, he aquí algunos ejemplos de plantillas básicas en ambos formatos:

En formato YAML:

```
1 Resources:
2   myBucket:
3     Type: 'AWS::S3::Bucket'
4   myDistribution:
5     Type: 'AWS::CloudFront::Distribution'
6     Properties:
7       DistributionConfig:
8         Origins:
9           - DomainName: !GetAtt
10             - myBucket
11             - DomainName
12             Id: myS3Origin
13             S3OriginConfig: {}
14         Enabled: 'true'
15         DefaultCacheBehavior:
16           TargetOriginId: myS3Origin
17           ForwardedValues:
18             QueryString: 'false'
19         ViewerProtocolPolicy: allow-all
```

En formato JSON:

```
1  {
2    "Resources": {
3      "myBucket": {
4        "Type": "AWS::S3::Bucket"
5      },
6      "myDistribution": {
7        "Type": "AWS::CloudFront::Distribution",
8        "Properties": {
9          "DistributionConfig": {
10           "Origins": [
11             {
12               "DomainName": {
13                 "Fn::GetAtt": [
14                   "myBucket",
15                   "DomainName"
16                 ]
17             },
18             "Id": "myS3Origin",
19             "S3OriginConfig": {}
20           ]
21         },
22         "Enabled": "true",
23         "DefaultCacheBehavior": {
24           "TargetOriginId": "myS3Origin",
25           "ForwardedValues": {
26             "QueryString": "false"
27           },
28           "ViewerProtocolPolicy": "allow-all"
29         }
30       }
31     }
32   }
33 }
34 }
```

Como Podemos apreciar, el formato YAML es más legible y fácil de entender, por lo que de media, es el más usado y el que más encontraremos en las distintas plantillas subidas a internet.

A continuación, voy a desglosar un poco más y explicar los distintos parámetros que podemos incluir en nuestra plantilla.

4.1 Desglose de una plantilla

4.1.1 Formato

Toda plantilla de AWS CloudFormation suele empezar con un breve texto que identifica la versión y el formato de la plantilla y, aunque opcional, es recomendado indicarlo para obedecer una correcta estructura en el código:

```
AWSTemplateFormatVersion: "2010-09-09"
```

4.1.2 Description

La sección de description, al igual que la anterior, es algo opcional. Pero es una buena practica incluirla siempre debido a que permite a quien lea el código a posteriori, saber con rapidez para qué sirve.

```
1 AWSTemplateFormatVersion: "2010-09-09"  
2 Description: >  
3   Here are some  
4   details about  
5   the template.
```

4.1.3 Metadata

Esta sección opcional permite al usuario introducir información relevante sobre la propia plantilla, como introducir detalles de recursos específicos que se introducirán más adelante.

```

1  AWSTemplateFormatVersion: "2010-09-09"
2  Description: >
3    Here are some
4    details about
5    the template.
6  Metadata:
7    Instances:
8      Description: "Information about the instances"
9    Databases:
10   Description: "Information about the databases"

```

A su vez, es en este mismo apartado donde, con la opción INIT, podremos introducir comandos y configuraciones necesarias para nuestras maquinas.

```

1  AWSTemplateFormatVersion: "2010-09-09"
2  Description: >
3    Here are some
4    details about
5    the template.
6  Metadata:
7    Instances:
8      Description: "Information about the instances"
9    Databases:
10   Description: "Information about the databases"
11   AWS::CloudFormation::Init:
12     configSets:
13       ascending:
14         - "config1"
15         - "config2"
16       descending:
17         - "config2"
18         - "config1"
19     config1:
20       commands:
21         test:
22           command: "echo \"${CFNTEST}\" > test.txt"
23           env:
24             CFNTEST: "I come from config1."
25           cwd: "~"
26     config2:
27       commands:
28         test:
29           command: "echo \"${CFNTEST}\" > test.txt"
30           env:
31             CFNTEST: "I come from config2"
32           cwd: "~"
33

```

4.1.4 Parameters

Los parámetros son otra sección opcional utilizada para “customizar” nuestras templates. Un ejemplo sería, especificar el tipo de memoria que queremos en nuestro sistema, o si queremos una imagen específica para una máquina virtual etc.

```
Parameters:
  InstanceTypeParameter:
    Type: String
    Default: t2.micro
    AllowedValues:
      - t2.micro
      - m1.small
      - m1.large
    Description: Enter t2.micro, m1.small, or m1.large. Default is t2.micro.
```

Una vez especificado el parámetro con un nombre de referencia (InstanceTypeParameter en este caso), podemos referenciarlo cuando creamos los recursos, haciendo uso de la fórmula *¡Ref* (Ejemplo en la sección *resources*)

4.1.5 Reglas

Nuevamente una sección opcional. Se utiliza para la validación de parámetros (O combinaciones de estos) que se pasan a una template durante la creación del stack o su actualización.

Esta sección, a pesar de ser opcional en la plantilla, es una de las más importantes a la hora de realizar un correcto mantenimiento de nuestros servicios y máquinas, dado que con estas reglas podremos evitar el error humano y de sintaxis en nuestra plantilla.

```

Rules:
  testInstanceType:
    RuleCondition: !Equals
      - !Ref Environment
      - test
    Assertions:
      - Assert:
          'Fn::Contains':
            - - a1.medium
            - !Ref InstanceType
          AssertDescription: 'For a test environment, the instance type must be a1.medium'
  prodInstanceType:
    RuleCondition: !Equals
      - !Ref Environment
      - prod
    Assertions:
      - Assert:
          'Fn::Contains':
            - - a1.large
            - !Ref InstanceType
          AssertDescription: 'For a production environment, the instance type must be a1.large'

```

Uno de los usos que más frecuentemente me he encontrado en los ejemplos encontrados mientras buscaba información, era el de asegurar que la plantilla era para producción y no desarrollo, o viceversa.

4.1.6 Mapping

Se trata de una sección opcional en la se pueden dar Aliases a los distintos nombres utilizados en la plantilla, con el fin de abreviar algunos de ellos.

```

Mappings:
  RegionMap:
    us-east-1:
      "HVM64": "ami-0ff8a91507f77f867"
    us-west-1:
      "HVM64": "ami-0bdb828fd58c52235"
    eu-west-1:
      "HVM64": "ami-047bb4163c506cd98"
    ap-southeast-1:
      "HVM64": "ami-08569b978cc4dfa10"
    ap-northeast-1:
      "HVM64": "ami-06cd52961ce9f0d85"

```

Esta opción es muy útil si vamos a tratar con numerosos IDs de resources, de manera que sean nombres simples e identificativos.

4.1.7 Condiciones

Las conditions o condiciones son un apartado opcional que contiene definidas las circunstancias en las que los distintos resources son creadas o configuradas. Por ejemplo, puedes crear una plantilla donde una resource solo se cree si cierta condición es verdadera.

Es similar a las reglas pero es más específica de las resources mientras que las reglas son algo general a la plantilla.

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  EnvType:
    Description: Environment type.
    Default: test
    Type: String
    AllowedValues:
      - prod
      - test
    ConstraintDescription: must specify prod or test.
Conditions:
  CreateProdResources: !Equals
    - !Ref EnvType
    - prod
```

4.1.8 Transforms

Se trata de una sección opcional de uso específico para macros que AWS CloudFormation utiliza para procesar la plantilla. Un ejemplo sería la transformación de los datos de un bucket a una Lambda.

4.1.9 Resources

La única sección obligatoria en un template, puesto que es donde se declaran que recursos procesará CloudFormation en su despliegue. (Por ejemplo, un EC2 o un S3 Bucket)

```
1  Resources:
2  MyEC2Instance:
3    Type: "AWS::EC2::Instance"
4  Properties:
5    ImageId: "ami-0ff8a91507f77f867"
6
```

4.1.10 Outputs

Sección opcional de la plantilla, donde uno puede declarar variables que luego serán exportadas para ser usadas en otras plantillas. Es importante recalcar que los valores exportados en esta sección, en caso de ser usados en otra plantilla, no podrán ser borrados (ni la plantilla tampoco) Deberán ser eliminados en escalera.

```
1  Outputs:
2  BackupLoadBalancerDNSName:
3    Description: The DNSName of the backup load balancer
4    Value: !GetAtt BackupLoadBalancer.DNSName
5    Condition: CreateProdResources
6  InstanceID:
7    Description: The Instance ID
8    Value: !Ref EC2Instance
```

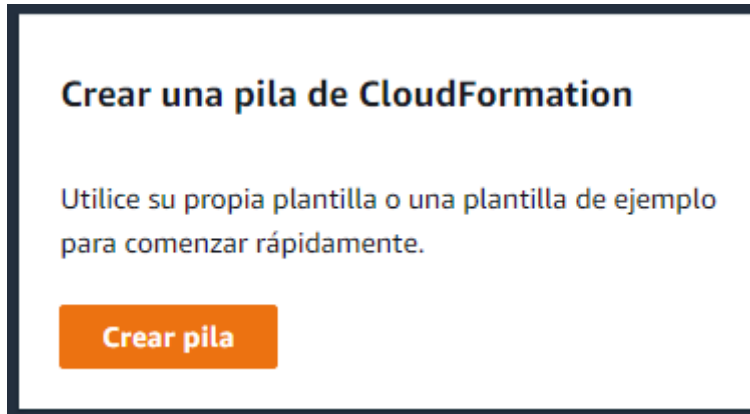
4.1.11 Pequeño resumen:

La conclusión de este apartado es que AWS CloudFormation nos proporciona muchos apartados de configuración y customización de nuestras plantillas, pudiendo incluso realizar plantillas múltiples que se comuniquen entre si.

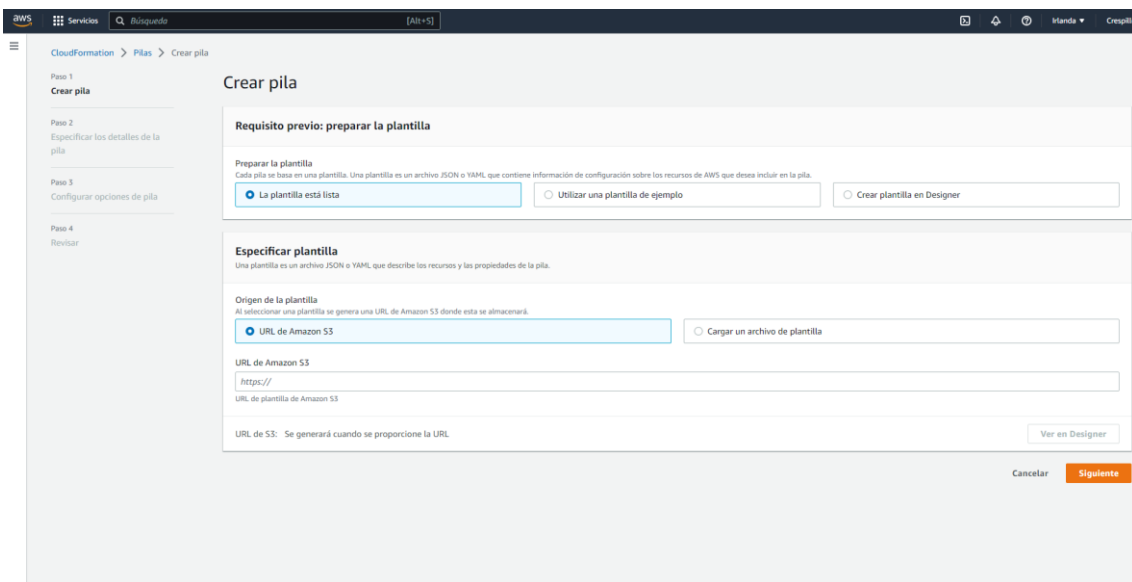
Lo más importante a recalcar de toda esta sección es que el único parámetro obligatorio en toda la plantilla es el de Resources. Todos los demás son opcionales.

5. Creación de un escenario simple de prueba

Comenzamos clicando en AWS CloudFormation dentro de nuestro panel de control AWS. Una vez dentro, veremos un botón denominado “Crear Pila”. Clicamos y entramos.



Una vez dentro, podremos observar que se nos ha desplegado un menú de creación de Pilas, con diferentes opciones. En mi caso, voy a crearme yo una plantilla a medida, por lo que selecciono “La plantilla está lista” Y dentro de esta sección, “Cargar un archivo de Plantilla”.



Ahora es el momento de, usando lo aprendido previamente, crear mi propia plantilla. En mi caso, voy a utilizar fragmentos de distintas plantillas de ejemplo para crear un LAMP personalizado, de forma que incluya muchos tipos de parámetros distintos para poder usarlos como ejemplo.

Esta es la plantilla que he usado.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "Esto es una template LAMP para ASIR IES GONZALO NAZARENO",

  "Parameters" : {

    "DBName": {
      "Default": "MyDatabase",
      "Description" : "MySQL Nombre de la Base de datos",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "64",
      "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription" : "Debe empezar con una letra y contener solo caracteres Alfanumericos."
    },

    "DBUser": {
      "NoEcho": "true",
      "Description" : "Username de la base de datos",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "16",
      "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription" : "Debe empezar con una letra y contener solo caracteres Alfanumericos."
    },

    "DBPassword": {
      "NoEcho": "true",
      "Description" : "Password de la base de datos",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "41",
      "AllowedPattern" : "[a-zA-Z0-9]*",
      "ConstraintDescription" : "Debe contener solo caracteres alfanumericos."
    },

    "DBRootPassword": {
      "NoEcho": "true",
      "Description" : "Contraseña root para la base de datos",
      "Type": "String",
      "MinLength": "1",
```



```

"MaxLength": "41",
"AllowedPattern": "[a-zA-Z0-9]*",
"ConstraintDescription": "Debe contener solo caracteres alfanumericos."
},

"InstanceType": {
  "Description": "Servidor Web",
  "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": "Debe ser un valor valido."
},

"SSHLocation": {
  "Description": "Ip desde la que se puede acceder a la instancia. Por defecto será 0.0.0.0",
  "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": "Tiene que ser una ip valida con el formato x.x.x.x/x."
}
},

"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" }
},

"AWSInstanceType2NATArch" : {
  "t1.micro" : { "Arch" : "NATHVM64" },
  "t2.nano" : { "Arch" : "NATHVM64" },
  "t2.micro" : { "Arch" : "NATHVM64" },
  "t2.small" : { "Arch" : "NATHVM64" },
  "t2.medium" : { "Arch" : "NATHVM64" },
  "t2.large" : { "Arch" : "NATHVM64" },
  "m1.small" : { "Arch" : "NATHVM64" },
  "m1.medium" : { "Arch" : "NATHVM64" },
```

```
"m1.large" : { "Arch" : "NATHVM64" },
"m1.xlarge" : { "Arch" : "NATHVM64" },
"m2.xlarge" : { "Arch" : "NATHVM64" },
"m2.2xlarge" : { "Arch" : "NATHVM64" },
"m2.4xlarge" : { "Arch" : "NATHVM64" },
"m3.medium" : { "Arch" : "NATHVM64" },
"m3.large" : { "Arch" : "NATHVM64" },
"m3.xlarge" : { "Arch" : "NATHVM64" },
"m3.2xlarge" : { "Arch" : "NATHVM64" },
"m4.large" : { "Arch" : "NATHVM64" },
"m4.xlarge" : { "Arch" : "NATHVM64" },
"m4.2xlarge" : { "Arch" : "NATHVM64" },
"m4.4xlarge" : { "Arch" : "NATHVM64" },
"m4.10xlarge" : { "Arch" : "NATHVM64" },
"c1.medium" : { "Arch" : "NATHVM64" },
"c1.xlarge" : { "Arch" : "NATHVM64" },
"c3.large" : { "Arch" : "NATHVM64" },
"c3.xlarge" : { "Arch" : "NATHVM64" },
"c3.2xlarge" : { "Arch" : "NATHVM64" },
"c3.4xlarge" : { "Arch" : "NATHVM64" },
"c3.8xlarge" : { "Arch" : "NATHVM64" },
"c4.large" : { "Arch" : "NATHVM64" },
"c4.xlarge" : { "Arch" : "NATHVM64" },
"c4.2xlarge" : { "Arch" : "NATHVM64" },
"c4.4xlarge" : { "Arch" : "NATHVM64" },
"c4.8xlarge" : { "Arch" : "NATHVM64" },
"g2.2xlarge" : { "Arch" : "NATHVMG2" },
"g2.8xlarge" : { "Arch" : "NATHVMG2" },
"r3.large" : { "Arch" : "NATHVM64" },
"r3.xlarge" : { "Arch" : "NATHVM64" },
"r3.2xlarge" : { "Arch" : "NATHVM64" },
"r3.4xlarge" : { "Arch" : "NATHVM64" },
"r3.8xlarge" : { "Arch" : "NATHVM64" },
"i2.xlarge" : { "Arch" : "NATHVM64" },
"i2.2xlarge" : { "Arch" : "NATHVM64" },
"i2.4xlarge" : { "Arch" : "NATHVM64" },
"i2.8xlarge" : { "Arch" : "NATHVM64" },
"d2.xlarge" : { "Arch" : "NATHVM64" },
"d2.2xlarge" : { "Arch" : "NATHVM64" },
"d2.4xlarge" : { "Arch" : "NATHVM64" },
"d2.8xlarge" : { "Arch" : "NATHVM64" },
"hi1.4xlarge" : { "Arch" : "NATHVM64" },
"hs1.8xlarge" : { "Arch" : "NATHVM64" },
"cr1.8xlarge" : { "Arch" : "NATHVM64" },
"cc2.8xlarge" : { "Arch" : "NATHVM64" }
}

"AWSRegionArch2AMI" : {
  "af-south-1" : {"HVM64" : "ami-064cc455f8a1ef504", "HVMG2" : "NOT_SUPPORTED"},
```

```

"ap-east-1"      : {"HVM64" : "ami-f85b1989", "HVMG2" : "NOT_SUPPORTED"},
  "ap-northeast-1"  : {"HVM64" : "ami-0b2c2a754d5b4da22", "HVMG2" : "ami-
09d0e0e099ecabba2"},
  "ap-northeast-2" : {"HVM64" : "ami-0493ab99920f410fc", "HVMG2" : "NOT_SUPPORTED"},
  "ap-northeast-3" : {"HVM64" : "ami-01344f6f63a4decc1", "HVMG2" : "NOT_SUPPORTED"},
  "ap-south-1"     : {"HVM64" : "ami-03cfb5e1fb4fac428", "HVMG2" : "ami-0244c1d42815af84a"},
  "ap-southeast-1"  : {"HVM64" : "ami-0ba35dc9caf73d1c7", "HVMG2" : "ami-
0e46ce0d6a87dc979"},
  "ap-southeast-2"  : {"HVM64" : "ami-0ae99b503e8694028", "HVMG2" : "ami-
0c0ab057a101d8ff2"},
  "ca-central-1"   : {"HVM64" : "ami-0803e21a2ec22f953", "HVMG2" : "NOT_SUPPORTED"},
  "cn-north-1"     : {"HVM64" : "ami-07a3f215cc90c889c", "HVMG2" : "NOT_SUPPORTED"},
  "cn-northwest-1" : {"HVM64" : "ami-0a3b3b10f714a0ff4", "HVMG2" : "NOT_SUPPORTED"},
  "eu-central-1"   : {"HVM64" : "ami-0474863011a7d1541", "HVMG2" : "ami-0aa1822e3eb913a11"},
  "eu-north-1"     : {"HVM64" : "ami-0de4b8910494dba0f", "HVMG2" : "ami-32d55b4c"},
  "eu-south-1"     : {"HVM64" : "ami-08427144fe9ebdef6", "HVMG2" : "NOT_SUPPORTED"},
  "eu-west-1"      : {"HVM64" : "ami-015232c01a82b847b", "HVMG2" : "ami-0d5299b1c6112c3c7"},
  "eu-west-2"      : {"HVM64" : "ami-0765d48d7e15beb93", "HVMG2" : "NOT_SUPPORTED"},
  "eu-west-3"      : {"HVM64" : "ami-0caf07637eda19d9c", "HVMG2" : "NOT_SUPPORTED"},
  "me-south-1"     : {"HVM64" : "ami-0744743d80915b497", "HVMG2" : "NOT_SUPPORTED"},
  "sa-east-1"      : {"HVM64" : "ami-0a52e8a6018e92bb0", "HVMG2" : "NOT_SUPPORTED"},
  "us-east-1"      : {"HVM64" : "ami-032930428bf1abbf", "HVMG2" : "ami-0aeb704d503081ea6"},
  "us-east-2"      : {"HVM64" : "ami-027cab9a7bf0155df", "HVMG2" : "NOT_SUPPORTED"},
  "us-west-1"      : {"HVM64" : "ami-088c153f74339f34c", "HVMG2" : "ami-0a7fc72dc0e51aa77"},
  "us-west-2"      : {"HVM64" : "ami-01fee56b22f308154", "HVMG2" : "ami-0fe84a5b4563d8f27"}
}

},

"Resources" : {

  "WebServerInstance": {
    "Type": "AWS::EC2::Instance",
    "Metadata" : {
      "AWS::CloudFormation::Init" : {
        "configSets" : {
          "InstallAndRun" : [ "Install", "Configure" ]
        },
      },

      "Install" : {
        "packages" : {
          "yum" : {
            "mysql"      : [],
            "mysql-server" : [],
            "mysql-libs" : [],
            "httpd"      : [],
            "php"        : [],
            "php-mysql"  : []
          }
        }
      }
    }
  }
}

```

```

},

"files" : {
  "/var/www/html/index.php" : {
    "content" : { "Fn::Join" : [ "", [
      "<html>\n",
      " <head>\n",
      "  <title>AWS CloudFormation Javier Crespillo ASIR</title>\n",
      "  <meta http-equiv=\"Content-Type\" content=\"text/html; charset=ISO-8859-1\">\n",
      " </head>\n",
      " <body>\n",
      "  <h1>Esto es un ejemplo de un LAMP</h1>\n",
      "  <p/>\n",
      "  <?php\n",
      "    // Print out the current data and time\n",
      "    print \"The Current Date and Time is: <br/>\";\n",
      "    print date(\"g:i A l, F j Y.\");\n",
      "  ?>\n",
      "  <p/>\n",
      "  <?php\n",
      "    // Setup a handle for CURL\n",
      "    $curl_handle=curl_init();\n",
      "    curl_setopt($curl_handle,CURLOPT_CONNECTTIMEOUT,2);\n",
      "    curl_setopt($curl_handle,CURLOPT_RETURNTRANSFER,1);\n",
      "    // Get the hostname of the instance from the instance metadata\n",
      "    curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/latest/meta-
data/public-hostname');\n",
      "    $hostname = curl_exec($curl_handle);\n",
      "    if (empty($hostname))\n",
      "    {\n",
      "      print \"Sorry, for some reason, we got no hostname back <br />\";\n",
      "    }\n",
      "    else\n",
      "    {\n",
      "      print \"Server = \" . $hostname . \"<br />\";\n",
      "    }\n",
      "    // Get the instance-id of the instance from the instance metadata\n",
      "    curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/latest/meta-
data/instance-id');\n",
      "    $instanceid = curl_exec($curl_handle);\n",
      "    if (empty($instanceid))\n",
      "    {\n",
      "      print \"Sorry, for some reason, we got no instance id back <br />\";\n",
      "    }\n",
      "    else\n",
      "    {\n",
      "      print \"EC2 instance-id = \" . $instanceid . \"<br />\";\n",
      "    }\n",
      "    $Database = \"localhost\";\n",

```

```

"   $DBUser   = \"", { "Ref" : "DBUser" }, "\";\n",
"   $DBPassword = \"", { "Ref" : "DBPassword" }, "\";\n",
"   print \"Database = \" . $Database . \"<br />\";\n",
"   $dbconnection = mysql_connect($Database, $DBUser, $DBPassword)\n",
"       or die(\"Could not connect: \" . mysql_error());\n",
"   print (\"Connected to $Database successfully\");\n",
"   mysql_close($dbconnection);\n",
"   ?>\n",
"   <h2>PHP Information</h2>\n",
"   <p/>\n",
"   <?php\n",
"   ?>\n",
"   </body>\n",
" </html>\n"
]],
"mode" : "000600",
"owner" : "apache",
"group" : "apache"
},

"/tmp/setup.mysql" : {
"content" : { "Fn::Join" : [ "", [
"CREATE DATABASE ", { "Ref" : "DBName" }, ";\n",
"GRANT ALL ON ", { "Ref" : "DBName" }, ".* TO ", { "Ref" : "DBUser" }, "@localhost
IDENTIFIED BY ", { "Ref" : "DBPassword" }, ";\n"
] ] },
"mode" : "000400",
"owner" : "root",
"group" : "root"
},

"/etc/cfn/cfn-hup.conf" : {
"content" : { "Fn::Join" : [ "", [
"[main]\n",
"stack=", { "Ref" : "AWS::StackId" }, "\n",
"region=", { "Ref" : "AWS::Region" }, "\n"
] ] },
"mode" : "000400",
"owner" : "root",
"group" : "root"
},

"/etc/cfn/hooks.d/cfn-auto-reloader.conf" : {
"content": { "Fn::Join" : [ "", [
"[cfn-auto-reloader-hook]\n",
"triggers=post.update\n",
"path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init\n",
"action=/opt/aws/bin/cfn-init -v ",
"    --stack ", { "Ref" : "AWS::StackName" },
"    --resource WebServerInstance ",

```

```

    "    --configsets InstallAndRun ",
    "    --region ", { "Ref" : "AWS::Region" }, "\n",
    "runas=root\n"
  ]],
  "mode" : "000400",
  "owner" : "root",
  "group" : "root"
}
},

"services" : {
  "sysvinit" : {
    "mysqld" : { "enabled" : "true", "ensureRunning" : "true" },
    "httpd" : { "enabled" : "true", "ensureRunning" : "true" },
    "cfn-hup" : { "enabled" : "true", "ensureRunning" : "true",
      "files" : [ "/etc/cfn/cfn-hup.conf", "/etc/cfn/hooks.d/cfn-auto-reloader.conf" ] }
  }
}
},

"Configure" : {
  "commands" : {
    "01_set_mysql_root_password" : {
      "command" : { "Fn::Join" : [ "", [ "mysqladmin -u root password '", { "Ref" : "DBRootPassword"
}, "" ] ] },
      "test" : { "Fn::Join" : [ "", [ "$mysql ", { "Ref" : "DBName" }, " -u root --password=", { "Ref"
: "DBRootPassword" }, "' >/dev/null 2>&1 </dev/null); (( $? != 0 ))" ] ] }
    },
    "02_create_database" : {
      "command" : { "Fn::Join" : [ "", [ "mysql -u root --password=", { "Ref" : "DBRootPassword" },
"' </tmp/setup.mysql" ] ] },
      "test" : { "Fn::Join" : [ "", [ "$mysql ", { "Ref" : "DBName" }, " -u root --password=", { "Ref"
: "DBRootPassword" }, "' >/dev/null 2>&1 </dev/null); (( $? != 0 ))" ] ] }
    }
  }
}
},
},
"Properties" : {
  "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
    { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" : "InstanceType" }, "Arch" ]
} ] },
  "InstanceType" : { "Ref" : "InstanceType" },
  "SecurityGroups" : [ { "Ref" : "WebServerSecurityGroup" } ],
  "UserData" : { "Fn::Base64" : { "Fn::Join" : [ "", [
    "#!/bin/bash -xe\n",
    "yum update -y aws-cfn-bootstrap\n",

    "# Install the files and packages from the metadata\n",

```

```

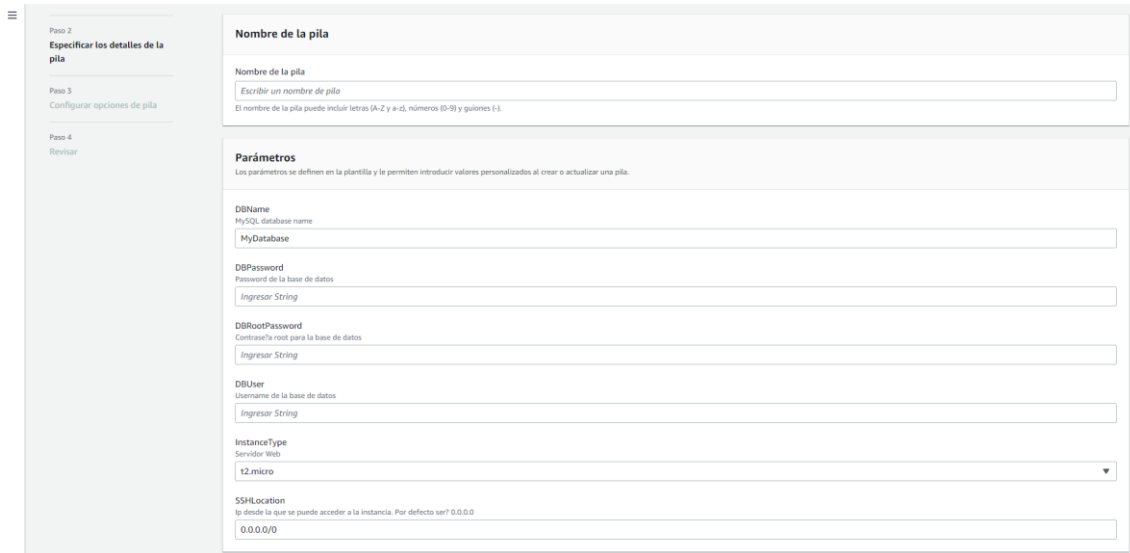
    "/opt/aws/bin/cfn-init -v ",
    "    --stack ", { "Ref" : "AWS::StackName" },
    "    --resource WebServerInstance ",
    "    --configsets InstallAndRun ",
    "    --region ", { "Ref" : "AWS::Region" }, "\n",

    "# Signal the status from cfn-init\n",
    "/opt/aws/bin/cfn-signal -e $? ",
    "    --stack ", { "Ref" : "AWS::StackName" },
    "    --resource WebServerInstance ",
    "    --region ", { "Ref" : "AWS::Region" }, "\n"
  ]}]
},
"CreationPolicy" : {
  "ResourceSignal" : {
    "Timeout" : "PT5M"
  }
}
},
"WebServerSecurityGroup" : {
  "Type" : "AWS::EC2::SecurityGroup",
  "Properties" : {
    "GroupDescription" : "Permite la conexion SSH/HTTP",
    "SecurityGroupIngress" : [
      { "IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "CidrIp" : "0.0.0.0/0"},
      { "IpProtocol" : "tcp", "FromPort" : "22", "ToPort" : "22", "CidrIp" : { "Ref" : "SSHLocation" }}
    ]
  }
}
},
"Outputs" : {
  "WebsiteURL" : {
    "Description" : "URL del lamp recientemente creado:",
    "Value" : { "Fn::Join" : [ "", [ "http://", { "Fn::GetAtt" : [ "WebServerInstance", "PublicDnsName" ] } ] ] }
  }
}
}
}
}
}

```

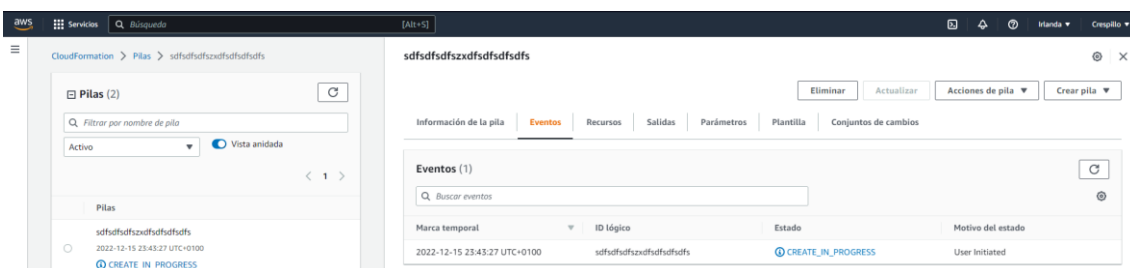
Una vez introducida la plantilla, procedemos con la siguiente página, donde se nos abrirán todas las posibles opciones de configuración que hemos introducido en nuestra plantilla.

Dado que estoy creando una plantilla para instalar un LAMP básico. No estoy haciendo que realice un aprovisionamiento automático sino simplemente que me de las opciones en caso de que yo quiera cambiarlas cada vez. Aunque si cambiáramos la sintaxis de la plantilla, podríamos hacerlo automáticamente sin necesidad de seleccionar nada.



Tras introducir todos estos campos procederemos con la siguiente pantalla de confirmación, donde veremos si los datos introducidos son correctos.

Una vez hecho esto, podemos proceder a esperar la creación de nuestro stack lamp



Marca temporal	ID lógico	Estado	Motivo del estado
2022-12-15 23:43:27 UTC+0100	sdfsdzdfszdfsdzdfsdzdfs	CREATE_IN_PROGRESS	User Initiated

CloudFormation > Pilas > sdfsfdsfzxdfsdfdsfdfs

Pilas (1)

sdffsfdsfzxdfsdfdsfdfs
2022-12-15 23:43:27 UTC+0100
● CREATE_COMPLETE

sdffsfdsfzxdfsdfdsfdfs

Información de la pila | **Eventos** | Recursos | Salidas | Parámetros | Plantilla | Conjuntos de cambios

Eventos (9)

Marca temporal	ID lógico	Estado	Motivo del estado
2022-12-15 23:44:53 UTC+0100	sdffsfdsfzxdfsdfdsfdfs	● CREATE_COMPLETE	-
2022-12-15 23:44:52 UTC+0100	WebServerInstance	● CREATE_COMPLETE	-
2022-12-15 23:44:51 UTC+0100	WebServerInstance	⌚ CREATE_IN_PROGRESS	Received SUCCESS signal with Uniqueid f-01732112b2b0ae9e4
2022-12-15 23:43:39 UTC+0100	WebServerInstance	⌚ CREATE_IN_PROGRESS	Resource creation Initiated
2022-12-15 23:43:36 UTC+0100	WebServerInstance	⌚ CREATE_IN_PROGRESS	-
2022-12-15 23:43:36 UTC+0100	WebServerSecurityGroup	● CREATE_COMPLETE	-
2022-12-15 23:43:35 UTC+0100	WebServerSecurityGroup	⌚ CREATE_IN_PROGRESS	Resource creation Initiated
2022-12-15 23:43:30 UTC+0100	WebServerSecurityGroup	⌚ CREATE_IN_PROGRESS	-
2022-12-15 23:43:27 UTC+0100	sdffsfdsfzxdfsdfdsfdfs	⌚ CREATE_IN_PROGRESS	User Initiated

Y si visitamos la sección de “Salidas”, podemos ver que tal y como configuramos los “Outputs” en nuestra plantilla, la url de la maquina estará ahí y podremos visitarla:

CloudFormation > Pilas > sdfsfdsfzxdfsdfdsfdfs

Pilas (1)

sdffsfdsfzxdfsdfdsfdfs
2022-12-15 23:43:27 UTC+0100
● CREATE_COMPLETE

sdffsfdsfzxdfsdfdsfdfs

Información de la pila | Eventos | Recursos | **Salidas** | Parámetros | Plantilla | Conjuntos de cambios

Salidas (1)

Clave	Valor	Descripción	Nombre de exportación
WebsiteURL	http://ec2-54-154-199-177.eu-west-1.amazonaws.com	URL del lamp recientemente creado:	-

Esto es un ejemplo de un LAMP

The Current Date and Time is:
10:46 PM Thursday, December 15 2022.

Server = ec2-54-154-199-177.eu-west-1.compute.amazonaws.com
EC2 instance-id = i-01732112b2b0ae9e4
Database = localhost
Connected to localhost successfully

PHP Information

Y con esto habríamos terminado la creación de un stack LAMP desde AWS Cloud Formation

6. Diferencias entre CloudFormation y otros IaC

Ahora que hemos descrito todos los parámetros principales que nos ofrece CloudFormation, y hemos hecho un escenario usando lo aprendido previamente, voy a pasar a describir un poco los puntos fuertes y débiles de CloudFormation frente a otros IaC como Terraform, que también son muy usados en el ámbito AWS.

En el caso de la comparativa de CloudFormation y Terraform ambas son herramientas de IaC, solo que la primera es desarrollada por AWS y solo sirve en el ecosistema de éste, mientras que la segunda fue desarrollada por Hashicorp y envuelve a muchos sistemas más allá de AWS.

En el caso de CloudFormation, se trata de una herramienta privada y no de código abierto, que sigue un roadmap dictaminado única y exclusivamente por amazon. Mientras tanto, Terraform es una herramienta opensource y su desarrollo lo dictamina una comunidad.

Terraform, como ya he mencionado previamente, funciona con una gran cantidad de sistemas y proveedores, es básicamente una navaja suiza que permite hacer de todo en casi todos los tipos de situaciones donde requiramos una herramienta del estilo. Esto implica que no es necesario aprender un lenguaje distinto cada vez que vamos a usar un proveedor de cloud diferente (AWS, azure...). A diferencia de esto, CloudFormation solo es usable en Amazon.

Dado su código abierto, el nivel de madurez de Terraform es mucho mayor al de cloudformation, por lo que puede decirse que posee mayor poder que CloudFormation.

Aun así, la rápida actualización y la gran cantidad de recursos de CloudFormation, hacen que la herramienta esté dirigida de forma muy precisa a lo que sus clientes necesitan, por lo que se actualiza a menudo de forma rápida y eficiente, rellenando los huecos en la programación.

En conclusión, nuestro objetivo con el uso de estas herramientas es conseguir infraestructuras robustas, con despliegues predecibles y fácilmente manejables mediante código.

Sin embargo, tener claras las virtudes y defectos de cada una de ellas es fundamental para conseguir este fin de una manera más eficiente y sencilla.

Cloudformation	Terraform
Closed Source, maintained/updated by AWS	Open source, many contributors
Suitable for working on AWS Cloud	Cloud Agnostic: Suitable for working with multi-cloud workloads
GUI access for no cost	GUI access requires expensive enterprise licence
No need to Manage State	Need to Manage State yourself
Supports YAML and JSON for configuration language	Supports JSON and HCL for configuration language
Nested Stacks lets you work with multiple templates. This concept is hard to grasp for beginners and has limitations	Working with multiple tf files easier .

7. Conclusiones, propuestas y dificultades

Como conclusión final de este proyecto, quiero expresar que todo lo explicado aquí no es nada más que una minúscula parte de lo que CloudFormation ofrece y puede ofrecer en el futuro.

Lo que yo he hecho no ha sido más que explicar los conceptos básicos de esta herramienta y su uso a través de la interfaz gráfica. Pero esta herramienta es mucho mas potente de lo demostrado aquí, sobre todo haciendo uso de su CLI, la cual no he tocado en este proyecto.

En mi opinión es una herramienta que puede llegar a ser tan útil como sus semejantes, y el hecho que sea solo de aws permite una mayor precisión y eficiencia en sus procesos.

Como propuesta para seguir profundizando, el uso de CLI junto a parámetros automáticos me parece la idea mas acertada para continuar el estudio de esta herramienta, así como las serverless applications y su despliegue, las cuales son la nueva moda en AWS.

Las dificultades encontradas en este proyecto han sido en su gran mayoría la propia documentación de AWS, que, aunque extensa, en ocasiones es excesivamente detallada y criptica por lo que no es nada amigable con el usuario novato. Durante el proceso de estudio de esta herramienta a menudo me he perdido en decenas de hojas de documentación explicando cosas distintas, pero con nombres similares y viceversa, por lo que ha retrasado mas de lo que me gustaría esta práctica.

8. Referencias

<https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/GettingStarted.html>