

Infrastructure as code

11 langues

Article Discussion

Lire Modifier Modifier le code Voir l'historique

Pour les articles homonymes, voir IAC.

L'**Infrastructure as code** (**IaC**) (littéralement : « infrastructure en tant que code ») est un ensemble de mécanismes permettant de gérer, par des fichiers descripteurs ou des **scripts** (code informatique), une infrastructure (informatique) virtuelle^{1,2}.

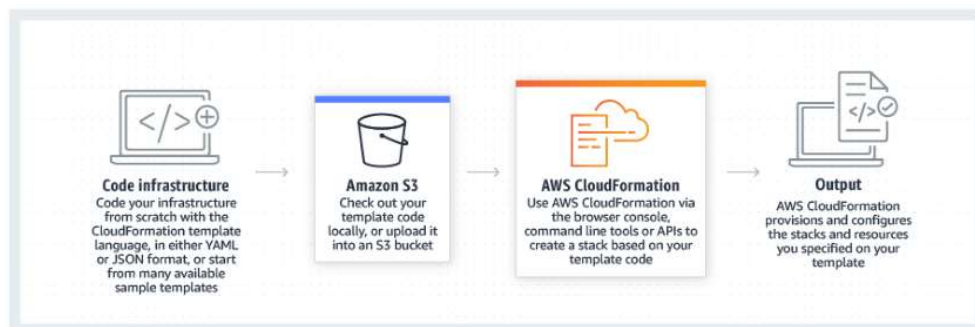
Initialement dédié aux **machines virtuelles** (également nommées « Instances »), l'évolution des offres dans le domaine de la **virtualisation** a rendu possible la gestion d'une infrastructure à part entière, de l'instance au **réseau**, incluant entre autres la gestion du service **DNS**, du « **Load-Balancing** », des **sous-réseaux** et des groupes de sécurité³.

Souvent plébiscité dans le cadre du **cloud computing**, l'Infrastructure as code offre aux **développeurs** la possibilité d'automatiser leurs déploiements de manière à éviter les tâches manuelles ou encore de devoir écrire d'eux-mêmes les appels aux **interfaces de programmation**. Cette technologie constitue une réponse aux besoins des entreprises en termes de **mise à l'échelle** des **applications** axée sur l'automatisation et la simplification de l'infrastructure de projets informatiques. De manière générale, l'Infrastructure as code s'inscrit dans la mouvance plus générale du **DevOps** qui a pour objectif d'unifier le développement logiciel et l'**administration système**.

AWS CloudFormation est un service qui vous permet de modéliser et de configurer vos ressources AWS de sorte que vous puissiez passer moins de temps à gérer ces ressources et consacrer plus de temps à vos applications exécutées dans AWS. Vous créez un modèle qui décrit toutes les ressources AWS que vous voulez (telles que des instances Amazon EC2 ou des instances de base de données Amazon RDS), et CloudFormation s'occupe de leur allocation et de leur configuration. Vous n'avez pas besoin de créer et de configurer individuellement les ressources AWS ni de déterminer leurs dépendances. CloudFormation se charge de tout. Les scénarios suivants montrent comment CloudFormation peut vous aider.

Fonctionnement

AWS CloudFormation vous permet de modéliser, de provisionner et de gérer les ressources AWS et tierces en traitant l'infrastructure en tant que code.



Cas d'utilisation

Gérer l'infrastructure avec DevOps

Automatiser, tester et déployer des modèles d'infrastructure avec des automatisations d'intégration et de livraison continues (CI/CD).

Mettre à l'échelle des piles de production

Exécuter tout ce que vous voulez, d'une seule instance Amazon Elastic Compute Cloud (EC2) à une application multi-régions complexe.

Partager les bonnes pratiques

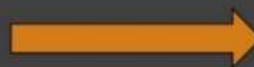
Définir facilement un sous-réseau Amazon Virtual Private Cloud (VPC) ou des services d'approvisionnement comme AWS OpsWorks ou Amazon Elastic Container Service (ECS).

Mots clés :

- Simplifie
- Tâches répétitives
- provisionnement

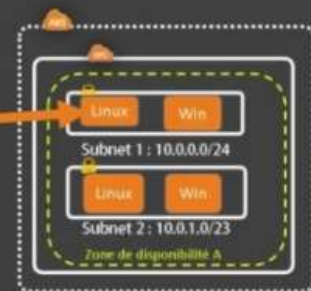


- AWS Management Console
- AWS CLI
- AWS SDK / API



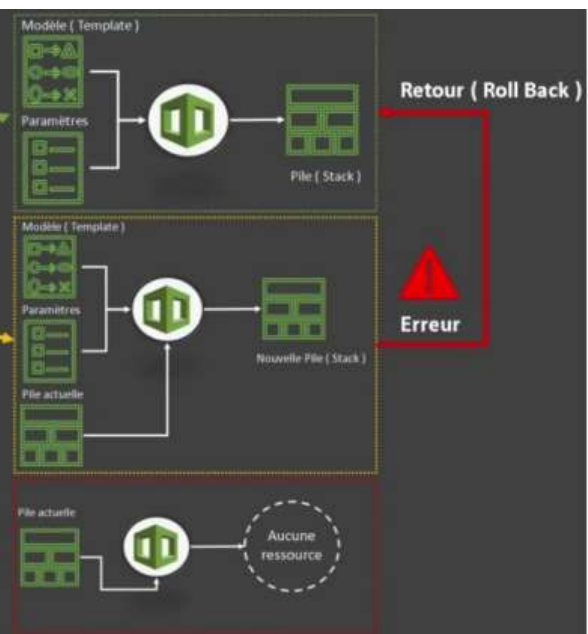
• Modèle (Template)

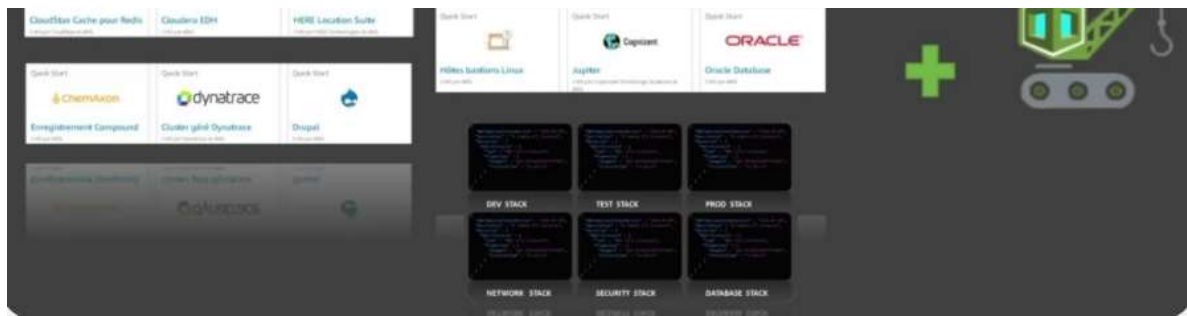
```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Description" : "A simple EC2 Instance",
  "Resources" : {
    "MyEC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : "ami-0ff8a91580777fb62",
        "InstanceType" : "t1.micro"
      }
    }
  }
}
```



Fonctionnalités

- Créer
- Mettre à jour
- Supprimer





Exemple de template :

<https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-ec2-instance.html>

Exemple de fichier de configuration :

YAML

```
Type: AWS::EC2::Instance
Properties:
  AdditionalInfo: String
  Affinity: String
  AvailabilityZone: String
  BlockDeviceMappings:
    - BlockDeviceMapping
  CpuOptions:
    CpuOptions
  CreditSpecification:
    CreditSpecification
  DisableApiTermination: Boolean
  EbsOptimized: Boolean
  ElasticGpuSpecifications:
    - ElasticGpuSpecification
  ElasticInferenceAccelerators:
    - ElasticInferenceAccelerator
  EnclaveOptions:
    EnclaveOptions
  HibernationOptions:
    HibernationOptions
  HostId: String
  HostResourceGroupArn: String
  IamInstanceProfile: String
  ImageId: String
  InstanceInitiatedShutdownBehavior: String
```

Les ressources présente dans ce fichier peuvent faire écho à des ressources existantes : ici on voit que le fichier de configuration propose les options "Subnet" et "SecurityGroup" à utiliser dans la création de notre EC2 :

```
SecurityGroupIds:
  - String
SecurityGroups:
  - String
SourceDestCheck: Boolean
SubnetId: String
```



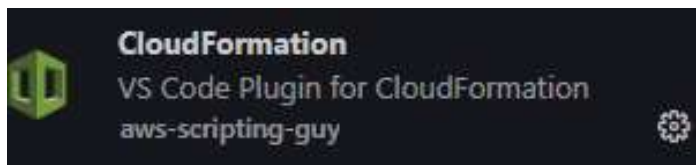
```

SsmAssociations:
  - SsmAssociation
SubnetId: String
Tags:
  - Tag
Tenancy: String
UserData: String
Volumes:
  - Volume

```

TP : générer un 1er fichier de configuration .yaml et le déployer

Pour nous faciliter notre développement, nous pouvons installer l'addon **CloudFormation** sur VSCode :



Capture d'écran de notre script yaml :

```

hello_world.json  ! ec2-instance.yaml X
! ec2-instance.yaml > {} Resources > {} benoitInstance
1  Resources:
2    benoitInstance:
3      Type: AWS::EC2::Instance
4      Properties:
5        ImageId: ami-06e0ce9d3339cb039
6        AvailabilityZone: eu-west-1a
7        InstanceType: t2.micro

```

Attention : l'AMI est différente en fonction des régions, bien s'assurer que nous sommes sur la bonne région quand on récupère l'AMI

De même, bien faire attention pour la zone de disponibilité : Il est important de définir une zone d'une région, par exemple: **eu-west-1a** ou **eu-west-2-a**

Pour ce build, on gardera uniquement les propriétés **ImageId**, **AvailabilityZone** et **InstanceType**

On va tester la création d'une stack avec notre build en rajoutant notre fichier :

Créer une pile

Prérequis - Préparer le modèle

Préparer le modèle

Chaque pile est basée sur un modèle. Un modèle est un fichier au format JSON ou YAML qui contient les informations de configuration sur les ressources AWS que vous souhaitez inclure dans la pile.

☒ Le modèle est prêt
☐ Utiliser un exemple de modèle
☐ Créer un modèle dans Designer

Spécifier un modèle

Un modèle est un fichier JSON ou YAML qui décrit les ressources et les propriétés de votre pile.

Source du modèle

La sélection d'un modèle génère un URL Amazon S3 où il sera stocké.

☐ URL Amazon S3
☒ Charger un fichier de modèle

Charger un fichier de modèle

ec2-instance.yaml

Fichier au format JSON ou YAML

URL S3 : https://s3.eu-west-1.amazonaws.com/cf-templates-hr04m3ov4urj-eu-west-1/2023-02-21T112825.811295r-ec2-instance.yaml

Spécifier les détails de la pile

Spécifier les détails de la pile

Nom de la pile

Nom de la pile

benoit-stack-test

Le nom de pile peut inclure des lettres (A-Z et a-z), des chiffres (0-9) et des tirets (-).

Paramètres

Les paramètres sont définis dans votre modèle et vous permettent de saisir des valeurs p

On va configurer notre stack : les parties **clé-valeur** sont **obligatoire**

Configurer les options de pile

Balises

Vous pouvez spécifier des balises (paires clé-valeur) à appliquer aux ressources de votre pile. Vous pouvez ajouter jusqu'à 50 balises uniques pour chaque pile.

Clé	Valeur - facultatif	
Q. ias_m2i	Q. test1	Supprimer
<input type="button" value="Ajouter une nouvelle balise"/>		

Vous pouvez ajouter 49 balise(s) supplémentaires(s).

Autorisations

Rôle IAM - facultatif

Choisissez le rôle IAM que CloudFormation utilisera pour toutes les opérations effectuées sur la pile.

Nom du rôle IAM

On peut également vérifier notre build :

Vérifier benoit-stack-test

Étape 1: Spécifier un modèle

Modèle

URL modèle

<https://s3.eu-west-1.amazonaws.com/cf-templates-hr04m3ov4urj-eu-west-1/2023-02-21T113045.851Zymo-ec2-instance.yaml>

Description de la pile

-

Second TP : création d'un script yaml qui génère une instance ec2 et un groupe de sécurité :

```
! ec2-instance-gs.yaml > {} Resources > {} SSHSecurityGroupBenoit > {} Properties > [ ] See
1 Resources:
2   BenoîtEC2Instance:
3     Type: AWS::EC2::Instance
4     Properties:
5       ImageId: ami-06e0ce9d3339cb039
6       InstanceType: t2.micro
7       AvailabilityZone: eu-west-1a
```

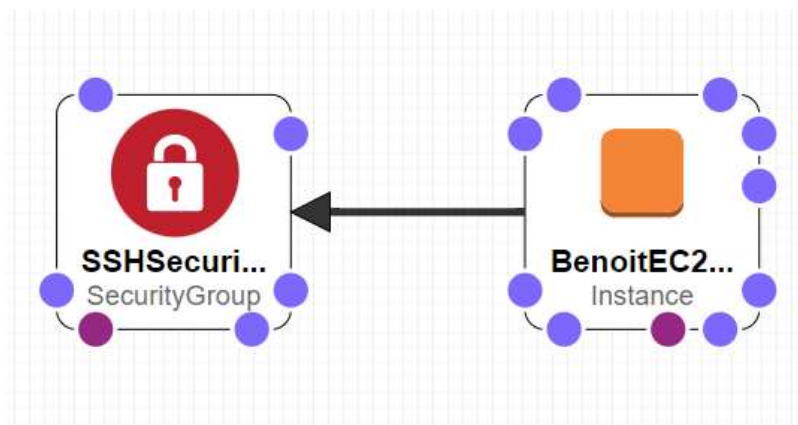
```

8     SecurityGroupIds:
9       - default
10      - !Ref SSHSecurityGroupBenoit
11  SSHSecurityGroupBenoit:
12    Type: AWS::EC2::SecurityGroup
13    Properties:
14      GroupName: gs-benoit-iac
15      GroupDescription: groupe de securite ssh pour exercice iac
16      SecurityGroupIngress:
17        - IpProtocol: tcp
18          FromPort: '22'
19          ToPort: '22'
20          CidrIp: 0.0.0.0/0

```

Le **!Ref** permet de faire référence à des ressources présente dans le fichier de configuration : syntaxe : **!Ref "nom_ressources"**

On peut vérifier dans le designer si tout a l'air bon :



Build de la stack :

Événements (8)				
Rechercher des événements				
Horodatage	ID logique	Statut	Motif du statut	
21-02-2023 14:19:56 UTC+0100	benoit-ecs-gs-ssh	CREATE_COMPLETE		
21-02-2023 14:19:54 UTC+0100	BenoitEC2Instance	CREATE_COMPLETE		
21-02-2023 14:19:02 UTC+0100	BenoitEC2Instance	CREATE_IN_PROGRESS	Resource creation Initiated	
21-02-2023 14:19:00 UTC+0100	BenoitEC2Instance	CREATE_IN_PROGRESS		
21-02-2023 14:18:59 UTC+0100	SSHSecurityGroupBenoit	CREATE_COMPLETE		
21-02-2023 14:18:58 UTC+0100	SSHSecurityGroupBenoit	CREATE_IN_PROGRESS	Resource creation Initiated	
21-02-2023 14:18:53 UTC+0100	SSHSecurityGroupBenoit	CREATE_IN_PROGRESS		
21-02-2023 14:18:49 UTC+0100	benoit-ecs-gs-ssh	CREATE_IN_PROGRESS	User Initiated	

Tp : création d'un script yaml qui génère une instance ec2, deux groupes se sécurité et une Elastic IP :

Code source de l'exercice :

```

! ec2-instance-gs-eip.yaml > {} Resources > {} BenoitEC2Instance > {} Properties > [ ] Secu
1  Resources:
2    BenoitEC2Instance:
3      Type: AWS::EC2::Instance
4      Properties:
5        ImageId: ami-06e0ce9d3339cb039
6        InstanceType: t2.micro
7        AvailabilityZone: eu-west-1a
8        SecurityGroupIds:
9          - !Ref SSHSecurityGroupBenoit
10         - !Ref ServerSecurityGroup
11      Tags:
12        - Key: Name
13          Value: benoit-instance

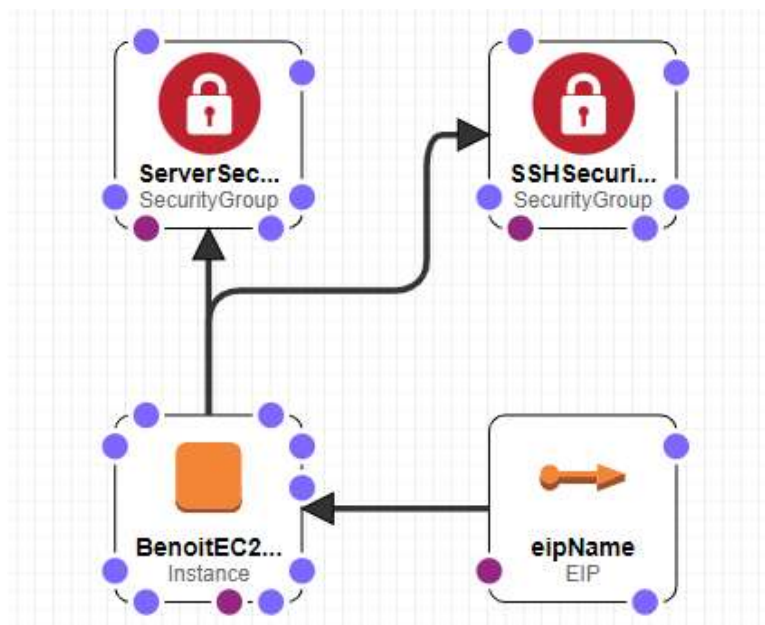
```

```

13     value: benoit-instance
14 SSHSecurityGroupBenoit:
15   Type: AWS::EC2::SecurityGroup
16   Properties:
17     GroupName: gs-benoit-iac
18     GroupDescription: groupe de securite ssh pour exercice iac
19     SecurityGroupIngress:
20       - IpProtocol: tcp
21         FromPort: '22'
22         ToPort: '22'
23         CidrIp: 0.0.0.0/0
24   ServerSecurityGroup:
25     Type: AWS::EC2::SecurityGroup
26     Properties:
27       GroupName: gs-benoit-web
28       GroupDescription: groupe de securite web
29       SecurityGroupIngress:
30         - IpProtocol: tcp
31           FromPort: '80'
32           ToPort: '80'
33           CidrIp: 0.0.0.0/0
34   eipName:
35     Type: AWS::EC2::EIP
36     Properties:
37       InstanceId: !Ref BenoitEC2Instance

```

Vérification du corde de designer :



Build de la stack :

Événements (14)			
Rechercher des événements			
Horodatage	ID logique	Statut	Motif du statut
21-02-2023 14:48:51 UTC+0100	benoit-stack-ec2-gs2-eip	CREATE_COMPLETE	-
21-02-2023 14:48:50 UTC+0100	eipName	CREATE_COMPLETE	-
21-02-2023 14:48:33 UTC+0100	eipName	CREATE_IN_PROGRESS	Resource creation Initiated
21-02-2023 14:48:31 UTC+0100	eipName	CREATE_IN_PROGRESS	-
21-02-2023 14:48:30 UTC+0100	BenoitEC2Instance	CREATE_COMPLETE	-
21-02-2023 14:47:59 UTC+0100	BenoitEC2Instance	CREATE_IN_PROGRESS	Resource creation Initiated
21-02-2023 14:47:57 UTC+0100	BenoitEC2Instance	CREATE_IN_PROGRESS	-
21-02-2023 14:47:56 UTC+0100	ServerSecurityGroup	CREATE_COMPLETE	-
21-02-2023 14:47:55 UTC+0100	SSHSecurityGroupBenoit	CREATE_COMPLETE	-
21-02-2023 14:47:55 UTC+0100	ServerSecurityGroup	CREATE_IN_PROGRESS	Resource creation Initiated
21-02-2023 14:47:55 UTC+0100	SSHSecurityGroupBenoit	CREATE_IN_PROGRESS	Resource creation Initiated
21-02-2023 14:47:50 UTC+0100	ServerSecurityGroup	CREATE_IN_PROGRESS	-
21-02-2023 14:47:50 UTC+0100	SSHSecurityGroupBenoit	CREATE_IN_PROGRESS	-
21-02-2023 14:47:47 UTC+0100	benoit-stack-ec2-gs2-eip	CREATE_IN_PROGRESS	User Initiated

Instance : i-Of7922837f1803443 (benoit-instance)

Détails
Sécurité
Mise en réseau
Stockage
Vérifications de statut
Surveillance
Balises

Résumé de l'instance Informations

ID d'instance
i-0f7922837f1803443 (benoit-instance)

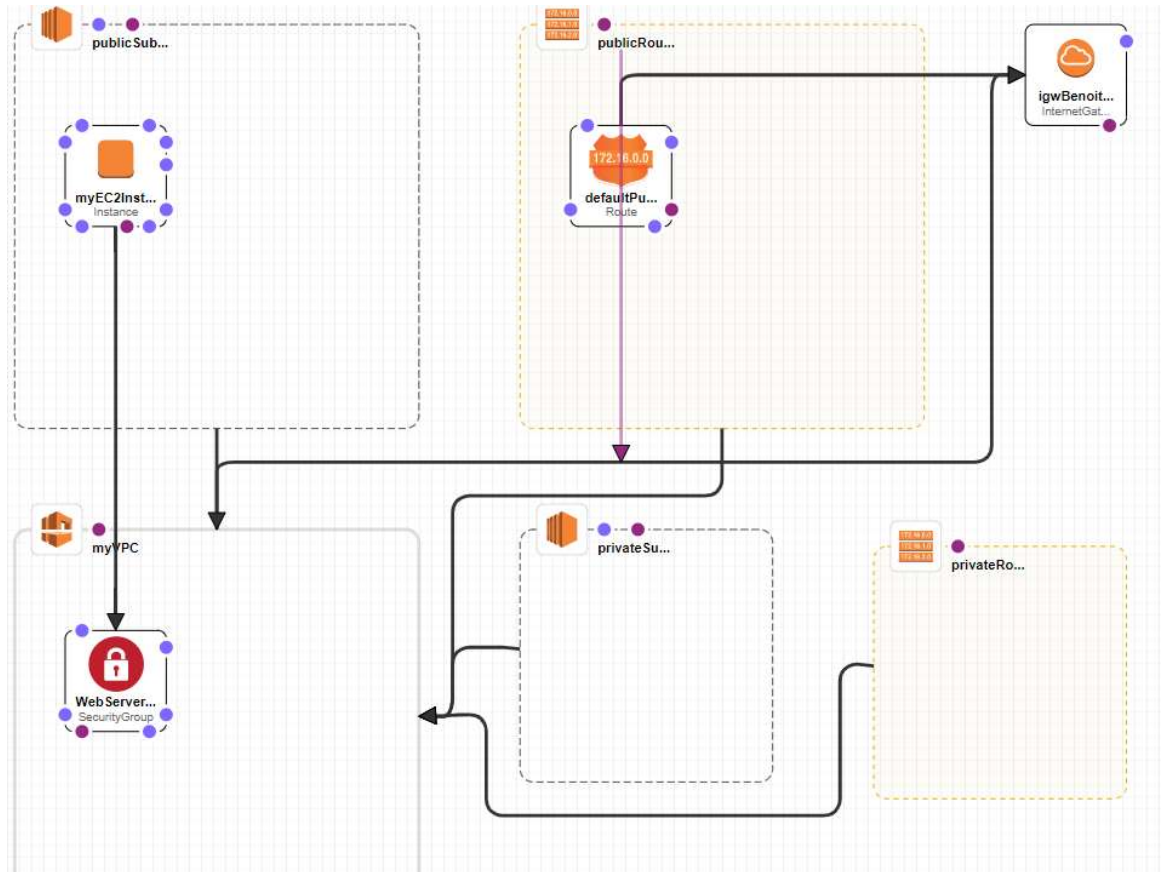
Adresse IPv4 publique
99.80.135.131 | [adresse ouverte](#)

Adresse IPv6
-

État de l'instance
En cours d'exécution

TP : création infrastructure VPC + EC2

https://docs.aws.amazon.com/fr_fr/AWSCloudFormation/latest/UserGuide/mappings-section-structure.html

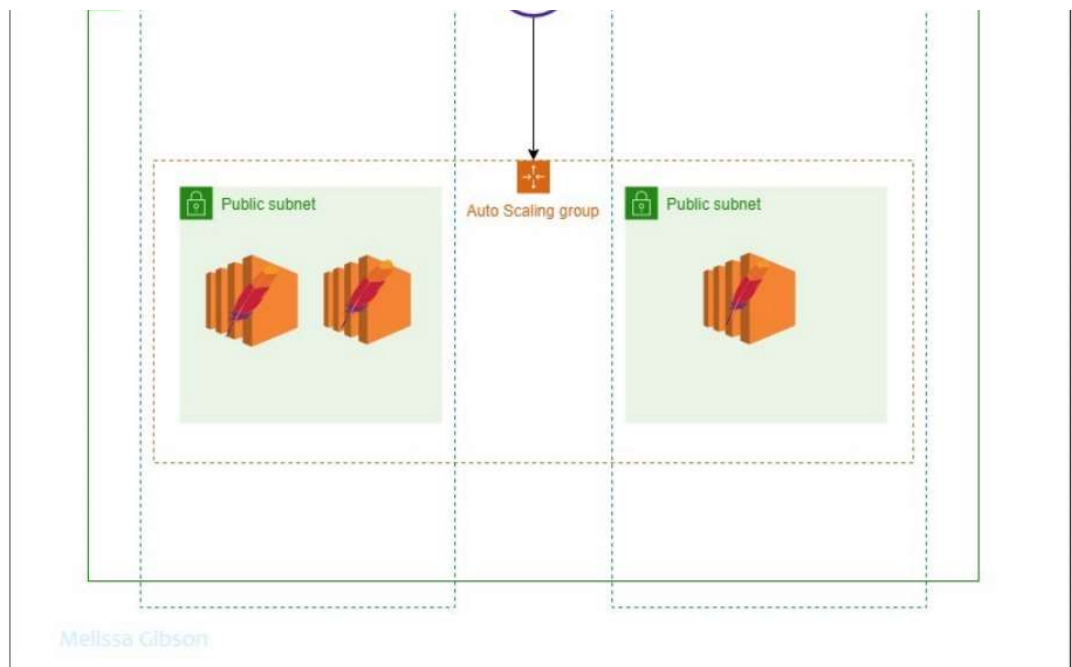


Horodatage	ID logique	Statut	Motif du statut
21-02-2023 16:53:42 UTC+0100	vpc-benoit	CREATE_COMPLETE	-
21-02-2023 16:53:41 UTC+0100	myEC2instance	CREATE_COMPLETE	-
21-02-2023 16:53:38 UTC+0100	defaultPublicRoute	CREATE_COMPLETE	-
21-02-2023 16:53:22 UTC+0100	defaultPublicRoute	CREATE_IN_PROGRESS	Resource creation initiated
21-02-2023 16:53:21 UTC+0100	defaultPublicRoute	CREATE_IN_PROGRESS	-
21-02-2023 16:53:20 UTC+0100	AttachGateway	CREATE_COMPLETE	-
21-02-2023 16:53:15 UTC+0100	routeTableAssocName	CREATE_COMPLETE	-
21-02-2023 16:53:15 UTC+0100	routeTableAssocName	CREATE_IN_PROGRESS	Resource creation initiated
21-02-2023 16:53:13 UTC+0100	routeTableAssocName	CREATE_IN_PROGRESS	-
21-02-2023 16:53:11 UTC+0100	publicRouteTable	CREATE_COMPLETE	-

Code source de ce TP disponible dans le dossier "Script" :

TP : réaliser cette infrastructure





Correction :

<https://towardsaws.com/creating-an-auto-scaling-group-in-aws-cloudformation-761b0f8c6364>

