

AWS FOR ALL

Ketsia Mulapi Tita



AWS POUR TOUS

partie 1 :

Exploration

Copyright ©2025 by Ketsia MULAPI

All rights reserved.

Contents

1	Structure et fonctionnalité	1
2	Terminologie	7
2.1	IAM Policy	7
2.2	Les couches d'identité et d'accès	9
2.3	Authentification et sécurité de connexion	17
2.4	Audit et supervision des accès	20
2.5	Interfaces d'administration	25
3	Modèle de responsabilité partagée	29
3.1	Modèle de responsabilité partagée	29
4	Facturation et Tarification	31
4.1	Facturation et tarification	31
5	Supervision et Conformité	33
5.1	Supervision et conformité	33
6	Services	35
6.1	Panorama des services AWS	35
6.2	Modèles de service du Cloud Computing	37

7 Documentation	41
8 Developer Resources	45
8.1 Developer Resources	45
9 Atelier	49

Chapter 1

Structure et fonctionnalité

AWS (**A**mazon **W**ebs **S**ervices) est la plateforme de cloud computing proposée par Amazon. Elle permet aux entreprises, développeurs et particuliers d'utiliser à distance des serveurs, du stockage, des bases de données, des outils d'intelligence artificielle, de la sécurité, et bien d'autres services ou ressources informatiques, sans avoir besoin d'acheter ni de gérer une infrastructure physique.

À titre d'exemple, plutôt que d'acheter un serveur, vous pouvez louer une machine virtuelle sur AWS via le service **EC2**, stocker des fichiers grâce au service **S3**, ou héberger votre application web avec **Lambda** ou **Elastic Beanstalk**.

Dans ce premier document, qui constitue le premier fascicule d'une série de plus de six, nous faisons un pas vers la découverte de l'écosystème AWS. Dans chaque document, la méthodologie employée repose sur **la pédagogie active et par le projet**, afin de favoriser une familiarisation rapide et concrète avec la plateforme. Pour commencer, rendez-vous sur <https://aws.amazon.com/fr/>

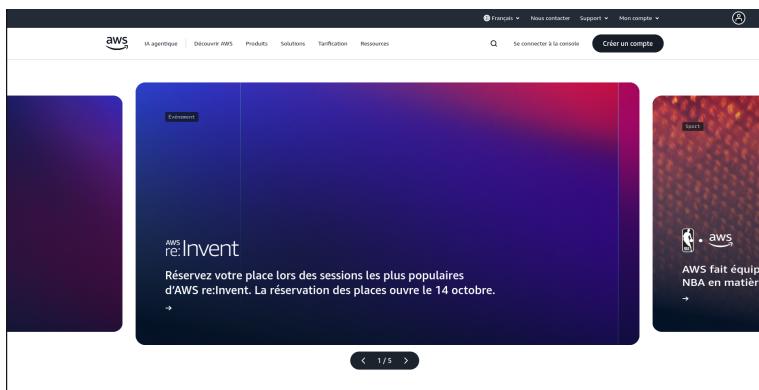


Figure 1.1: Page d'accueil AWS

1. **Navigation principale :** Sert de point d'accès global à toutes les pages AWS.
2. **Aux sections clés :** Comprendre où chercher l'information selon le besoin.
3. **Carrousel visuel :** Les campagnes et actualités majeures d'AWS, avec la dynamique d'innovation et les grands événements communautaires (ici **AWS re:Invent**).

Infrastructure mondiale AWS

Le Cloud AWS couvre 120 Zones de disponibilité dans 38 Régions géographiques, avec 10 Zones de disponibilité et 3 Régions AWS supplémentaires à venir dans les régions suivantes : le Royaume d'Arabie saoudite, Chili et le Cloud souverain européen AWS.

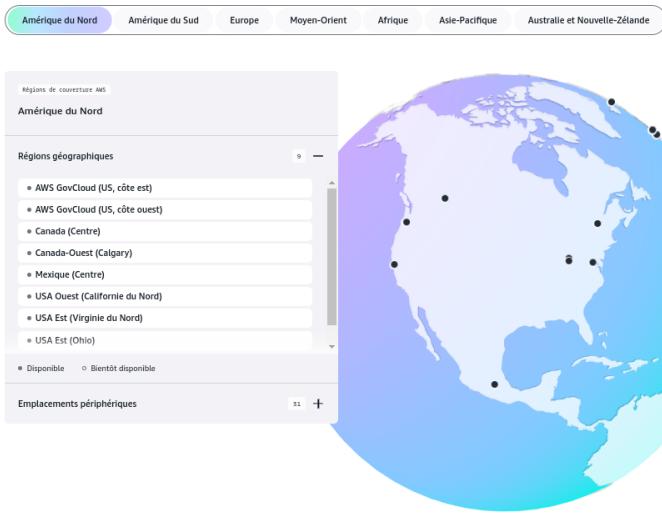


Figure 1.2: Infrastructure mondiale d'AWS

L'infrastructure d'Amazon Web Services (AWS) repose sur un modèle distribué constitué de **Régions** et de **Zones de disponibilité (AZ)**.

- 1. 38 Régions géographiques** réparties dans le monde.
- 2. 120 Zones de disponibilité (AZ)** : isolées mais interconnectées, assurant la **tolérance aux pannes**, la **redondance** et la **haute disponibilité**.
- 3. Régions en expansion** pour répondre aux besoins de déploiement localisés selon la **latence**, les **contraintes réglementaires** ou la **proximité des utilisateurs** — notamment en **Arabie Saoudite**, au **Chili**, et avec le **Cloud Souverain Européen** conforme au **RGPD**.

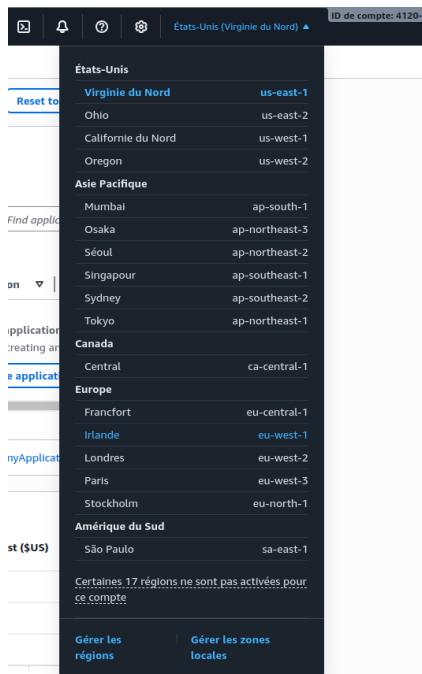


Figure 1.3: Choix et gestion des régions AWS : choix de l’Irlande

Vous devez toujours vérifier votre région AWS avant de travailler

- **Où ?** La région AWS se sélectionne directement dans la **console AWS**, en haut à droite, à côté du nom d’utilisateur.

- **Quand ?** Le *choix de la région* s’effectue :

1. avant la création d’une ressource (machine virtuelle, base de données, bucket S3, etc.), car la plupart des services AWS sont **régionaux** et ne peuvent pas être déplacés après création ;
2. ou lors d’un changement de région dans la console, pour gérer des ressources situées ailleurs.

- Comment et pourquoi choisir ?

Le choix d'une région dépend de plusieurs critères :

1. **Proximité géographique** : réduire la latence en choisissant la région la plus proche des utilisateurs ;
2. **Conformité et législation** : respecter les obligations locales (ex. **RGPD** en Europe) ;
3. **Disponibilité des services** : tous les services ne sont pas disponibles dans toutes les régions ;
4. **Coût** : certaines régions sont légèrement moins chères que d'autres (ex. *US East*) ;
5. **Résilience** : utiliser plusieurs régions pour assurer la **tolérance aux pannes**.

Pour consulter la disponibilité des services par région :

<https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/>

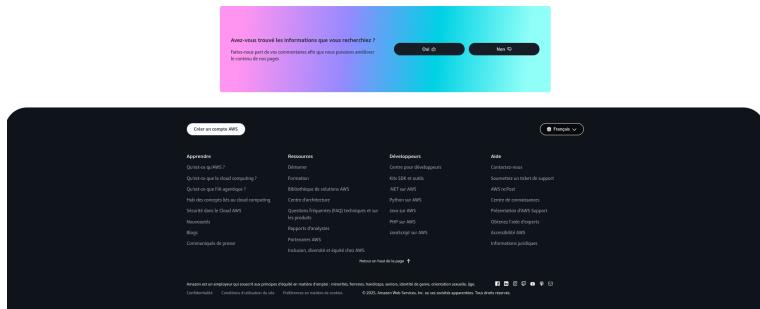


Figure 1.4: plus bas ...

- **Apprendre** : Définitions, nouveautés, sécurité, blog.
- **Ressources** : Documentation, FAQ, centre d'architecture.
- **Développeurs** : Kits SDK (*.NET, Python, PHP, Java, JS*).
- **Aide** : Support, accessibilité, informations juridiques.

Chapter 2

Terminologie

AWS est un univers riche en terminologie. Pour mieux le comprendre, établissons un parallèle entre un **compte AWS** et un **bâtiment** : chaque identité, rôle ou ressource occupe un espace bien défini, avec des règles d'accès précises.

2.1 IAM Policy

IAM ou Identity and Access Management(Gestion des identités et des accès), c'est un concept qui décrit qui (user) a le droit de faire quoi (limites, rôles, privilèges) une fois connecté :

1. **Gestion** : les utilisateurs humains ou application, les groupes, les rôles, les politiques
2. **Surveillance (Access Analyzer) et Audit (CloudTrail)** : qui s'est connecté ? qui a fait quoi ?

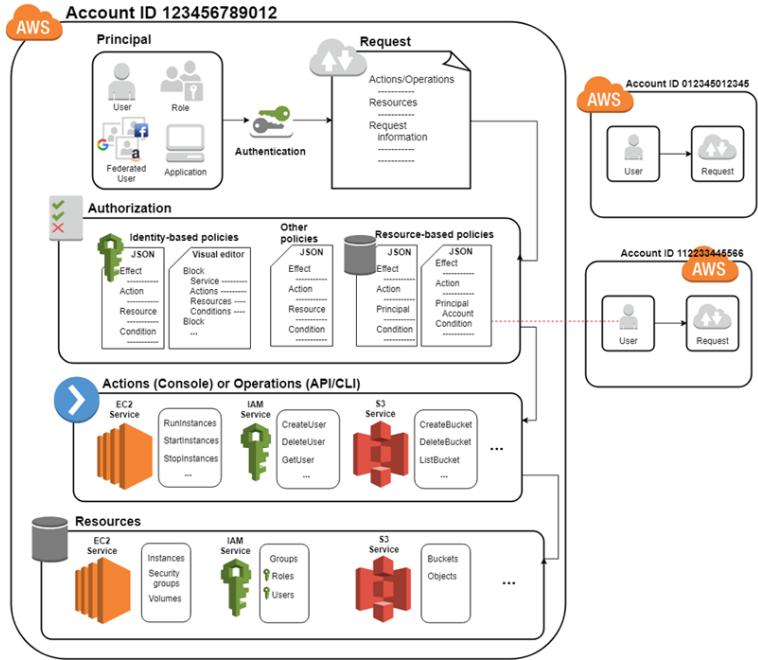


Figure 2.1: How IAM works

1. L'utilisateur (humain ou application) s'authentifie via ses identifiants.
2. IAM identifie le *principal* (user, rôle, identité fédérée) et valide sa légitimité.
3. Une demande d'accès est envoyée à IAM.
4. IAM évalue les politiques d'autorisation et accorde ou refuse l'accès.
5. Toutes les actions sont consignées pour audit.

2.2 Les couches d'identité et d'accès

Profil (AWS Builder ID)

C'est une méthode votre identité personnelle (indépendante d'un compte IAM)

1. Utile pour accéder aux services communautaires : **formations, forums, certifications.**
2. Permet la gestion des MFA, sessions et paramètres de confidentialité.
3. **Vous, en tant qu'individu dans l'écosystème AWS (hors infrastructure).**

C'est tout simplement votre profil que vous pouvez trouver ici : <https://profile.aws.amazon.com//profile/details>

Compte AWS (root or Root Account)

Ce n'est pas toujours l'administrateur, c'est simplement le propriétaire du compte AWS créé à l'inscription AWS, il est lié à une adresse email (pro/perso) et un mot de passe (pwd) (root credentials)

1. Détient **tous les droits absolus** (facturation, suppression, fermeture de compte).
2. Doit être protégé par une **MFA**.
3. À utiliser uniquement pour la gestion, jamais pour les opérations quotidiennes(EC2, S3, etc).
4. **Le propriétaire du bâtiment.**

Administrateur

C'est l'utilisateur IAM qui dispose de la politique **AdministratorAccess**.

1. Personne (souvent un membre du service informatique et/ou utilisateur IAM avec la politique **AdministratorAccess**) qui gère un compte AWS.
2. Crée les utilisateurs IAM, définit les rôles et les politiques.
3. Gère les accès, MFA, alias et configurations SSO.
4. **Le gestionnaire qui remet les clés et fixe les règles**

Utilisateur (ou Utilisateur IAM)

C'est une identité permanente, qui peut être **une personne ou application** autorisée à accéder à AWS

1. Chaque utilisateur IAM possède ses propres identifiants et permissions(distincts du root).
2. Peut accéder uniquement aux ressources définies dans ses politiques. (ex : accès à EC2, S3)
3. **L'utilisateur final qui exécute les opérations autorisées (autorisé à opérer dans le bâtiment.).**

Rôle IAM

C'est une identité temporaire sans identifiants fixes, utilisée par **des services ou comptes tiers**, pour exécuter une tâche spécifique (principe du “least privilege”).

Un rôle IAM est attachée à une politique d'autorisation, laquelle est un triplet (Service, Action, Ressource).

1. Repose sur le principe du *least privilege*
2. **Comparable à un badge temporaire pour accéder à une pièce précise.**

Un exemple; on rattache au rôle r la politique p suivante : pour une ressource **DynamoDB**, je ne veux pas accorder un accès *read-only* général (qui permettrait de lire toute la table ou des listes d'éléments), mais uniquement autoriser la lecture d'un seul élément, sous condition de son identifiant (ID).

- **Utilisateur IAM** : utilisateur créé dans AWS, qui n'a pas forcément de rôle par défaut.
- **Rôle r** : entité qui assume les permissions et que l'on attribue à un utilisateur.
- **Policy p** : document (JSON,...) décrivant les actions et conditions autorisées, attaché au rôle.
- **Action** : dynamodb:GetItem (et non dynamodb:Scan ou dynamodb:Query) ;
- **Condition** : filtrer sur un PartitionKey ou un Item ID donné.

Distinction entre User IAM et Role IAM

1. Quand créer un User IAM

Un **User IAM** représente une **personne humaine** (ou un compte applicatif unique) qui se connecte directement à AWS pour exécuter des actions via :

- la **console AWS** (identifiant + mot de passe) ;
- la **CLI AWS** avec Access Key ID et Secret Access Key ;
- des **outils ou scripts externes** (par exemple, une plate-forme CI/CD).

Exemples :

- Un administrateur : `admin-lead`
- Un développeur : `dev1`
- Une application externe : `github-actions-user`

Les permissions d'un **User IAM** sont permanentes et définies par les **policies (autorisations)** qui lui sont directement attachées.

2. Quand créer un Role IAM

Un **Role IAM** est une identité temporaire attribuée à :

- un service AWS (EC2, Lambda, SageMaker, etc.) ;

- un autre user ou compte AWS (via la commande `assume-role`).

Un rôle contient :

- une **policy d'autorisation** (ce que le rôle peut faire) ;
- une **trust policy** (qui a le droit d'assumer ce rôle).

Exemples :

- **EC2S3BackupRole** : permet à une instance EC2 d'envoyer des fichiers vers S3.
- **SageMakerRole** : permet à un data scientist d'accéder à SageMaker.
- **LambdaDynamoRole** : permet à une fonction Lambda de lire dans DynamoDB.

3. Comparatif synthétique

Objectif d'accès	Type
humain	User IAM
service AWS	Role IAM
user	Role IAM assumé

Objectif d'accès	Connexion directe
humain	Oui
service AWS	Non
user	Oui (via STS)

Objectif d'accès	Durée
humain	Permanente
service AWS	Temporaire
user	Temporaire (1h à 12h)

Objectif d'accès	Exemple
humain	admin1, dev1
service AWS	EC2S3BackupRole
user	SageMakerRole

4. Donner des permissions temporaires à un User IAM existant

Pour accorder des permissions temporaires à un utilisateur déjà existant :

1. Créer un rôle avec les permissions souhaitées :

```
aws iam create-role \
--role-name S3TempAccessRole \
--assume-role-policy-document \
file://trust-policy.json
```

2. Définir la **trust policy** (dans `trust-policy.json`) :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {"AWS":
        "arn:aws:iam::123456789012:user/dev1"},
      "Action": "sts:AssumeRole"
    }
  ]
}
```

3. Attacher les permissions nécessaires au rôle :

```
aws iam attach-role-policy \
--role-name S3TempAccessRole \
--policy-arn
arn:aws:iam::aws:policy/AmazonS3FullAccess
```

4. Le user lui-même exécute ensuite la commande suivante pour qu'il assume ce rôle :

```
aws sts assume-role \
--role-arn
arn:aws:iam::123456789012:role
/S3TempAccessRole \
--role-session-name sessionTemporaire
```

5. Cela lui renvoie des **credentials temporaires** (AccessKey, SecretKey, Token) valides pour une durée limitée (1h par défaut, jusqu'à 12h maximum).

Ces clés temporaires peuvent ensuite être utilisées dans un profil distinct :

```
aws configure --profile session-temp
# puis exécuter les commandes avec :
aws s3 ls --profile session-temp
```

2.3 Authentification et sécurité de connexion

AWS Identity Center (ex-SSO ou, ex-AWS SSO)

C'est le système d'authentification centralisé pour gérer ou orchestré l'accès à plusieurs comptes AWS avec une seule identité :

1. Remplace l'ancien service AWS SSO (Single-Sign-On)
2. Géré par l'administrateur AWS
3. Permet la fédération d'identités (connexion avec identifiants professionnels)
4. Il permet l'attribution de groupes et permissions centralisées
5. Intègre CloudTrail pour la journalisation des connexions.
6. C'est donc le pont entre votre identité en entreprise et votre compte AWS
7. **C'est comment vous vous connectez ? C'est votre moyen d'entrer dans le bâtiment. ?**

Identifiants de l'entreprise

Les ID de l'entreprise sont reliés à “l’**AWS Identity Center**”

1. Il permet aux employés d'une société de se connecter à AWS avec leurs identifiants professionnels
2. Il est géré par le service informatique de l'entreprise
3. **C'est votre identité dans l'entreprise, votre badge personnel. Qui êtes-vous au sein du bâtiment ?**

Alias de compte (Account Alias)

C'est l'**URL personnalisée** de connexion IAM.

1. Nom lisible qui remplace l'ID numérique dans l'URL de connexion.
2. (UN exemple
<https://alias.signin.aws.amazon.com/console>) au lieu de l'ID numérique
3. **L'adresse d'entrée vers ton environnement AWS (l'adresse du bâtiment).**

Identifiant de compte AWS (Account ID)

1. **ID unique à 12 chiffres** du compte (ex.: 123456789012).
2. Utilisé dans les **politiques IAM**, les **rôles inter-comptes**, et pour la **facturation**.
3. **Comparable au numéro d'un appartement/pièce.**

Informations d'identification du compte root

Identifiants initiaux (e-mail + mot de passe) créés à l'inscription du compte AWS.

1. **Le root a toujours tous les droits** sur le compte, à **réservé** à la facturation, à la gestion du compte et à la récupération d'accès (pas pour l'opérationnel quotidien).
2. Ajouter impérativement une **MFA** au root.
3. *Clés d'accès (Access key/Secret key)* : **déconseillées pour le root** (préférer des utilisateurs IAM ou des rôles).
4. **Les clés maîtresses du propriétaire du bâtiment.**

MFA (Multi-Factor Authentication)

C'est le code temporaire envoyé (SMS, application d'authentification, clé U2F) pour valider une connexion sécurisée.

1. Recommandée pour **tous les comptes sensibles** (root, admins, comptes à priviléges).
2. **Le verrou/sur-verrou de sécurité (double des clés).**

2.4 Audit et supervision des accès

AWS CloudTrail

Le CloudTrail répertorie les connexions, c'est un service qui peut :

1. Journaliser les **événements** (ConsoleLogin, API, IAM, SSO) des utilisateurs **root**, **IAM** et **fédérés**.
2. Traçabilité par **région** et stockage centralisable pour audit et conformité.
3. **Le registre des entrées/sorties (les caméras)**.

IAM Access Analyzer

1. Analyse les **autorisations effectives** (qui a accès à quoi).
2. Déetecte les partages **non intentionnels** (publics ou inter-comptes).
3. **Le détective/gardien de la sécurité IAM.**

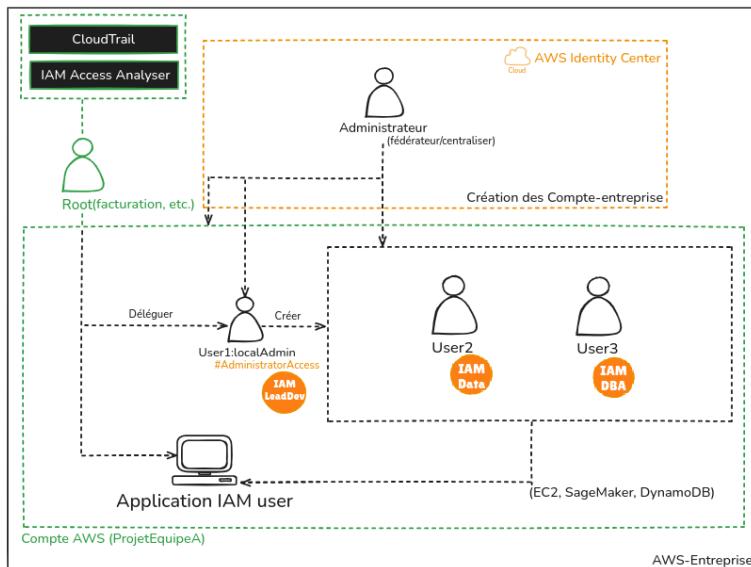


Figure 2.2: AWS en entreprise

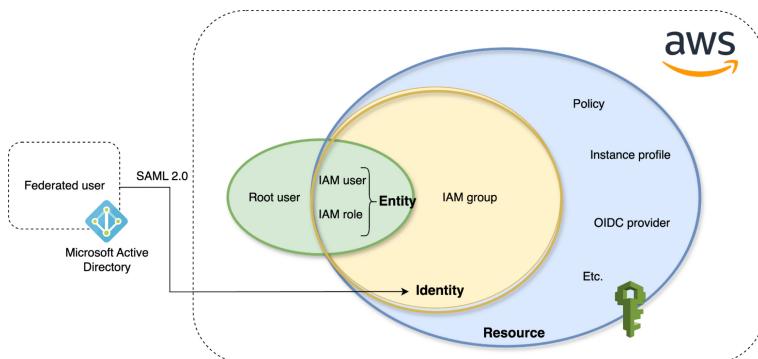


Figure 2.3: Composantes clées AWS pour IAM

Concrètement : IAM et gestion hiérarchique des accès

Principe fondamental Maintenant que vous savez que le service **AWS Identity and Access Management (IAM)** permet de définir et de contrôler qui peut accéder aux ressources, quelles actions il peut exécuter et à partir de quel service. Chaque entité AWS agit à travers une identité IAM : un **utilisateur**, un **rôle** ou une **application**.

1. Le compte root : autorité initiale , est l'identité suprême de l'organisation AWS. Il doit uniquement servir à :

- créer le premier utilisateur IAM d'administration ;
- gérer la facturation et la configuration organisationnelle;
- jamais à créer ni gérer directement les ressources ou les profils d'équipe.

Erreur fréquente : créer des utilisateurs ou des ressources avec le compte Root. Cela brise la hiérarchie de sécurité et empêche la traçabilité des actions.

2. Création du profil administrateur IAM À partir du compte Root, on crée un premier utilisateur IAM (souvent nommé **AdminLead**) auquel on attache la politique gérée par AWS :

Policy: AdministratorAccess

Ce profil dispose de permissions complètes sur les services AWS. C'est lui qui servira de point d'entrée pour créer les autres profils.

3. Connexion via IAM pour la gestion des utilisateurs

Pour respecter la hiérarchie :

1. Connectez-vous avec le profil IAM disposant du rôle **AdministratorAccess** ;
2. Créez les autres utilisateurs (ex. **Dev**, **DataAnalyst**, **BackupOperator**) ;
3. Créez ensuite des rôles spécifiques pour chaque usage.

Ainsi, le compte Root reste isolé, et la gestion quotidienne se fait via IAM.

4. Structure logique

```
Root Account → AdminLead (AdministratorAccess)
                  User2: Dev (gestion EC2)
                  User3: DataAnalyst (accès DataBase)
                  User4: BackupOperator (sauvegarde
→          S3)
```

5. Rôles et services AWS

: Un **rôle IAM** est une identité temporaire attribuée à un utilisateur ou un service. Les services AWS (comme une instance EC2, un Lambda, etc.) ne peuvent pas avoir de clés permanentes. Pour leur permettre d'interagir avec d'autres services (S3, DynamoDB, CloudWatch...), il faut leur attacher un rôle.

```
# Exemple : attacher un rôle à une instance EC2
Role: S3BackupRole
Policy: accès au bucket s3://sauvegarde-app
Trust policy: autorise le service EC2 à assumer le
→ rôle
```

Une fois le rôle attaché, l'instance EC2 peut exécuter :

```
aws s3 cp /data/fichier.txt s3://sauvegarde-app/
```

sans qu'aucune clé d'accès ne soit stockée localement sur la machine.

6. Donc en conclusion pour respecter la hiérarchie

- Le compte Root crée uniquement le profil **AdministratorAccess**.
- Le profil Administrateur crée les autres profils IAM, les services et les rôles pour les profils et services.
- Les rôles définissent les permissions et les relations entre utilisateurs et services.

Cette séparation garantit la sécurité, la traçabilité et la conformité dans une architecture AWS bien structurée.

2.5 Interfaces d'administration

Console de gestion AWS

1. Interface web (*Management Console*) pour gérer et surveiller les services (EC2, S3, RDS, etc.).

AWS CLI / SDK / IaC (CDK)

1. **CLI** pour automatiser via terminal ;
SDK (.NET, Python, PHP, Java, JS) pour intégrations applicatives.
2. Alternatives programmatiques à la Console pour l'intégration et le déploiement **CI/CD** et **IaC**(Infrastructure as Code, ex: usage du *Cloud Development Kit (CDK)*).

Schéma logique : de l'identité à l'autorisation

Avez-vous compris ?

1. **Qui êtes-vous ?** Identité (Profil AWS Builder ID, Identity Center/SSO, IAM).
2. **Comment vous connectez-vous ?** Authentification (Root/IAM/SSO Credentials, MFA, Alias).
3. **Que pouvez-vous faire ?** Autorisation (Policies IAM, Rôles, Groupes).
4. **Qui gère ?** Administration (Administrateur IAM, Propriétaire/root).
5. **Qui a fait quoi ?** Audit (CloudTrail) & Supervision (Access Analyzer).

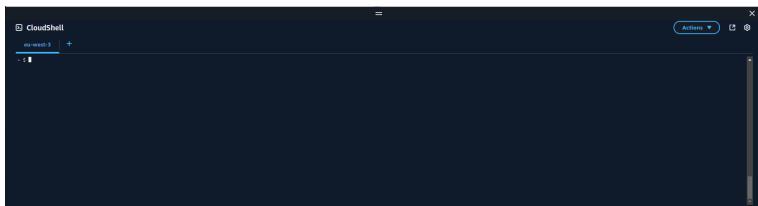


Figure 2.4: Vue de la CLI sous AWS

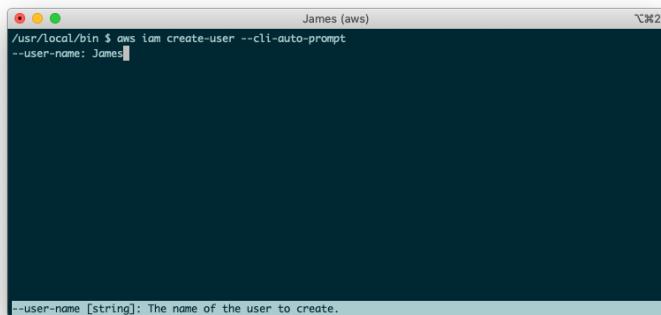


Figure 2.5: Vue de la CLI sous macOS

Installation AWS CLI sur votre poste de travail

Rendez-vous sur ce lien et en fonction de votre OS, choisissez le mode d'installation :

https://docs.aws.amazon.com/fr_fr/cli/latest/userguide/getting-started-install.html

Nous verrons comment interagir avec la console et la CLI.

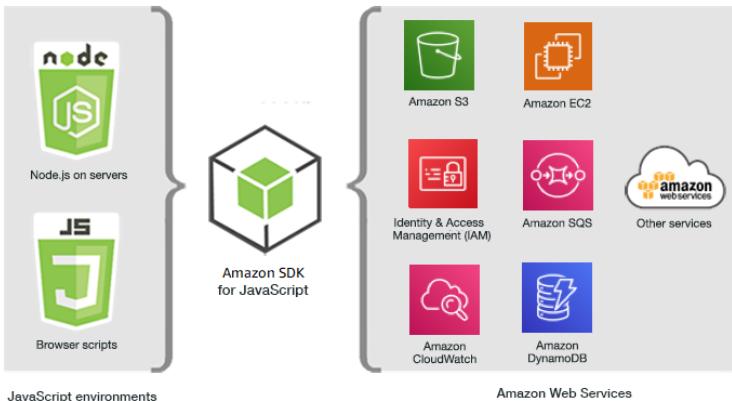


Figure 2.6: AWS SDK

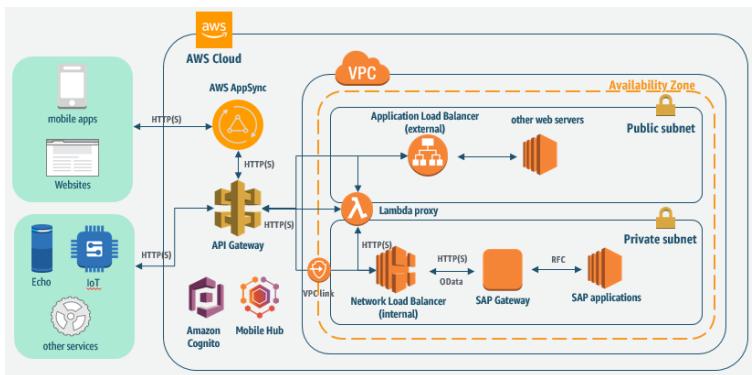


Figure 2.7: API

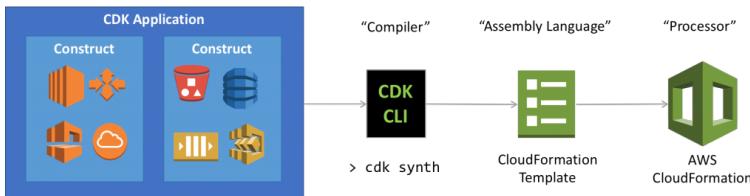


Figure 2.8: Modélisation fonctionnelle du CDK

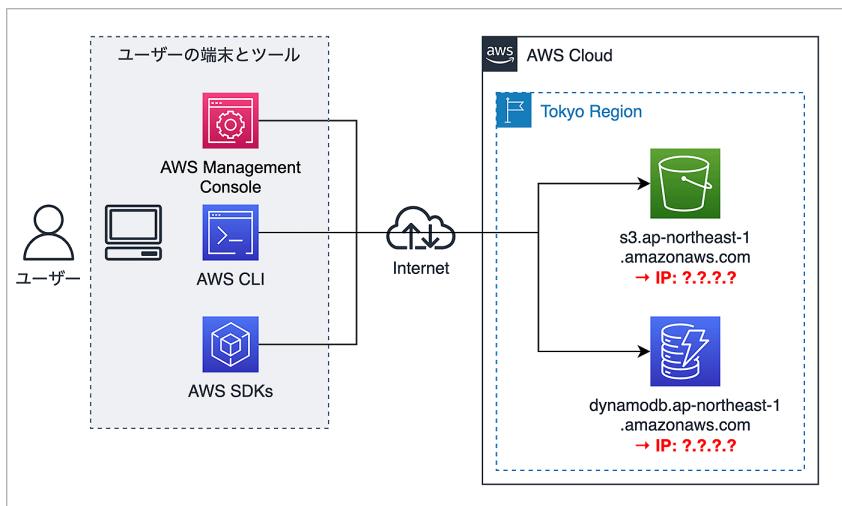


Figure 2.9: Les différentes approches d'utilisation d'AWS (AWS forum)

Chapter 3

Modèle de responsabilité partagée

3.1 Modèle de responsabilité partagée

AWS repose sur un modèle de **responsabilité partagée** entre **AWS** et le **client**. Cela signifie que la sécurité du cloud est gérée conjointement :

1. Axé sur le matériel (hardware), **AWS** est responsable de la **sécurité du cloud** (infrastructures , réseau, disponibilité).
2. **Le client** est responsable de la **sécurité dans le cloud** (données et configuration, software).

AWS protège le bâtiment, vous sécurisez ce qu'il y a à l'intérieur.

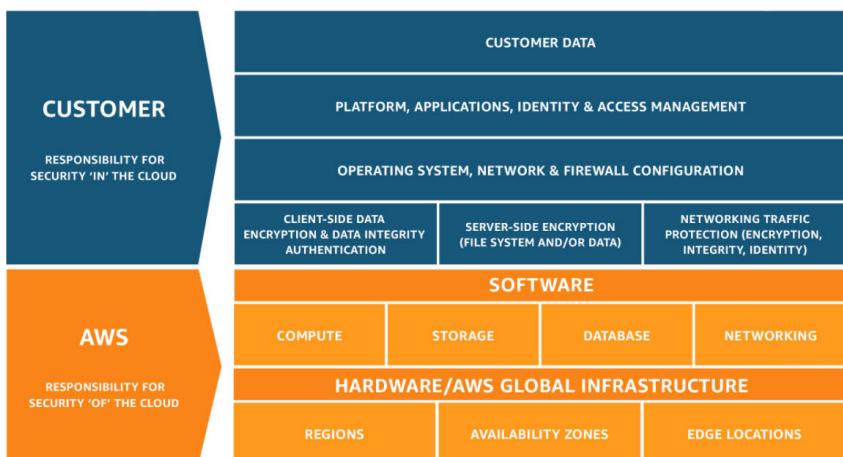


Figure 3.1: Les responsabilités partagées

Chapter 4

Facturation et Tarification

4.1 Facturation et tarification

Principe général

AWS fonctionne selon un modèle de **facturation à l'usage** (pay-as-you-go) :

1. Vous payez uniquement ce que vous consommez.
2. La planification budgétaire est essentielle pour éviter les dépassements.
3. Les coûts peuvent croître rapidement sans suivi précis.

Modèles économiques

1. **On-Demand** (par défaut) : paiement à l'heure ou à la seconde selon l'usage réel.
2. **Reserved Instances** : réservation d'instances sur 1 ou 3 ans avec réduction de coût (Tolérantes aux interruptions).
3. **Spot Instances** : utilisation de capacité non utilisée à prix réduit, mais interrompable.

Outils de gestion des coûts

1. **AWS Pricing Calculator** : estimation des coûts avant déploiement.
2. **AWS Budgets** : suivi et alertes budgétaires personnalisées.
3. **AWS Cost Explorer** : visualisation, analyse et optimisation des dépenses.

Chapter 5

Supervision et Conformité

5.1 Supervision et conformité

Surveillance et performance

avec **Amazon CloudWatch**

1. Surveillance en temps réel des performances et de la disponibilité.
2. Création d'alarmes et tableaux de bord personnalisés.

Audit et traçabilité

avec **AWS CloudTrail**

1. Journalisation des connexions et appels API.
2. Suivi des activités IAM, SSO et Root pour la conformité et la sécurité.

Optimisation et bonnes pratiques

avec AWS Trusted Advisor

1. Recommandations automatiques pour :

- (a) la **sécurité**,
- (b) la **performance**,
- (c) la **réduction des coûts**.

Nous venons de voir dans les chapitre 3, 4 et 5, que AWS fournit les outils ; la responsabilité d'en faire bon usage vous appartient car, maîtriser vos coûts, c'est maîtriser votre architecture.

Chapter 6

Services

6.1 Panorama des services AWS

AWS propose un vaste **catalogue de plus de 200 services** couvrant tous les besoins d'une infrastructure cloud moderne. Ces services sont organisés en **familles logiques** selon leur rôle dans une architecture cloud.

Principe général

- Le catalogue AWS évolue continuellement : de nouveaux services apparaissent chaque année.
- Chaque service appartient à une **catégorie fonctionnelle** (Compute, Réseau, Stockage, etc.).
- L'objectif est de permettre une architecture **modulaire et scalable**, où chaque composant répond à un besoin précis.

AWS ne se limite pas à des serveurs : c'est un écosystème complet (Cloud as a Lego).

Catégories essentielles à connaître

- **Compute** : exécution d'applications, serveurs virtuels et conteneurs (EC2, Lambda, ECS, EKS).
- **Réseau (Networking)** : connectivité, équilibre de charge et DNS (VPC, Route 53, CloudFront, ELB).
- **Stockage (Storage)** : gestion et persistance des données (S3, EBS, EFS, Glacier).
- **Bases de données (Databases)** : relationnelles ou NoSQL (RDS, Aurora, DynamoDB, Redshift).
- **Sécurité et identité** : gestion des accès et conformité (IAM, KMS, Cognito, GuardDuty).
- **Surveillance et observabilité (Monitoring)** : supervision et alertes (CloudWatch, CloudTrail, X-Ray).
- **Déploiement et automatisation (DevOps)** : CI/CD et gestion d'infrastructure (CloudFormation, CodePipeline, Elastic Beanstalk).

Domaines spécialisés

- **Intelligence Artificielle et Machine Learning** (Sage-Maker, Rekognition, Comprehend, Bedrock).
- **Analytique et Big Data** (Athena, EMR, Glue, Kinesis, QuickSight).
- **IoT et Edge Computing** (IoT Core, Greengrass, Snowball).
- **Sécurité avancée et gouvernance** (Security Hub, Macie, Organizations).

- Migration et intégration hybride (DMS, SnowFamily, Outposts, Direct Connect).

6.2 Modèles de service du Cloud Computing

Cette distinction nous permet de situer le **positionnement d’AWS** dans les principaux modèles de service du Cloud Computing. Chaque modèle représente un degré différent de gestion et de responsabilité entre AWS et le client. Vous verrez que plus on monte dans le modèle, moins on gère l’infrastructure mais on dépend de plus en plus du fournisseur

IaaS – Infrastructure as a Service

- AWS fournit les **ressources fondamentales** : serveurs virtuels, stockage, et réseaux.
- Le client installe, configure et gère le système d’exploitation, les applications et les données.
- **Exemples :** EC2 (machines virtuelles), EBS (stockage en bloc), VPC (réseau virtuel privé).
- **Vous louez le bâtiment vide et aménagez l’intérieur.**

PaaS – Platform as a Service

- AWS gère à la fois l’infrastructure et la plateforme logicielle.
- Le client déploie directement ses applications sans gérer les serveurs sous-jacents.

- **Exemple :** Elastic Beanstalk (déploiement automatisé d'applications).
- **Vous entrez dans un bâtiment déjà équipé et prêt à accueillir vos applications.**

SaaS – Software as a Service

- AWS ou ses partenaires fournissent des **applications prêtes à l'emploi**, accessibles via Internet.
- L'utilisateur n'a plus à gérer ni infrastructure, ni maintenance, ni mise à jour.
- **Exemples :** Amazon WorkMail (messagerie d'entreprise), Amazon Chime (visioconférence).
- **Vous utilisez le service sans jamais gérer le bâtiment ni l'installation.**

Services by category	
Calcul	Outils pour développeurs
EC2 Lambda Batch Amazon Beanstalk Serverless Application Repository AWS Outposts AWS Compute Layer AWS App Runner AWS Lambda@Edge AWS Lambda@Edge Service AWS Lambda View	CodeCommit CodeBuild CodeDeploy CodePipeline CloudWatch CloudShell KMS AWS FIS Infrastructure Composer AWS App Studio AWS Amplify CodeArtifact Amazon CodeCatalyst Amazon CodePipeline (Including Amazon CodeWhisperer)
Conteneur	Habiliter le client
Amazon Container Service Amazon Kubernetes Service Amazon Lambda@Edge Service on AWS Amazon Container Registry	AWS IoT Managed Services Amazon Device SDKs AWS IoT Core Private Support AWS Blockchain Amazon Managed Blockchain
Stockage	Machine Learning
S3 EFS FSx S3 Glacier Storage Gateway AWS Backup AWS Config Reprise après sinistre d'AWS Amazon S3	Amazon SageMaker AI Amazon SageMaker AI Assistant Amazon Codestar Amazon DevOps Guru Amazon Forecast Amazon Fraud Detector Amazon Personalize Amazon Personnel Amazon Polly Amazon Rekognition Amazon Rekognition Amazon Transcribe Amazon Translate AWS DeepRacer AWS Personalize Amazon Marketplace AWS HealthLake Amazon Lookout for Vision Amazon Lookout for Equipment Amazon Lookout for Metrics Amazon IoT Greengrass Amazon IoT Greengrass Amazon Greengrass Amazon Greengrass Amazon Comprehend Medical Amazon Lex Amazon Rekognition
Base de données	Gestion financière dans le cloud
Aurora et RDS Amazon DocumentDB Amazon Keyspaces Amazon Redshift DynamoDB Aurora ESQL Amazon MercuryDB Oracle Database@AWS	AWS Billing AWS Billing Connector Gestion de la facturation et des coûts
Migration et transfert	Mobile
AWS Migration Hub AWS Application Migration Service Amazon Database Migration Service Amazon Transfer Family AWS Snow Family DataSync AWS Transfer AWS Transform AWS Migration and Modernization Amazon Elastic Weaver Service	AWS Amplify AWS AmplifySync Device Farm Amazon Location Service
Mise en réseau et diffusion de contenu	Intégration des applications
VPC CloudFront API Gateway Direct Connect AWS App Mesh Global Accelerator Route 53 AWS Data Transfer Terminal AWS Cloud Map Application Recovery Controller	Step Functions Amazon AppFlow Amazon MQ Simple Notification Service Simple Queue Service SWF Amazon Airline gréé AWS B2B Data Interchange Amazon EventBridge
Services par type	Applications d'entreprise
	Amazon Connect Amazon Device Farm Amazon Simple Email Service Amazon WorkDocs Amazon WorkMail AWS Supply Chain Amazon Pinpoint Amazon One Enterprise AWS WAF AWS AppFabric AWS Lambda Packaging Amazon Chime SDK
	Computing de l'utilisateur final
	WorkSpaces AppStream WorkSpaces Thin Client WorkSpaces Web Browser
	Internet des objets
	IoT Analytics IoT Device Defender IoT Device Management IoT Events Manager IoT Sterilize IoT Core IoT Meshmaker IoT Events AWS IoT FleetWise
	Développement de jeux
	Amazon GameLift Servers Amazon GameLift Streams

Figure 6.1: Catalogue des Services AWS

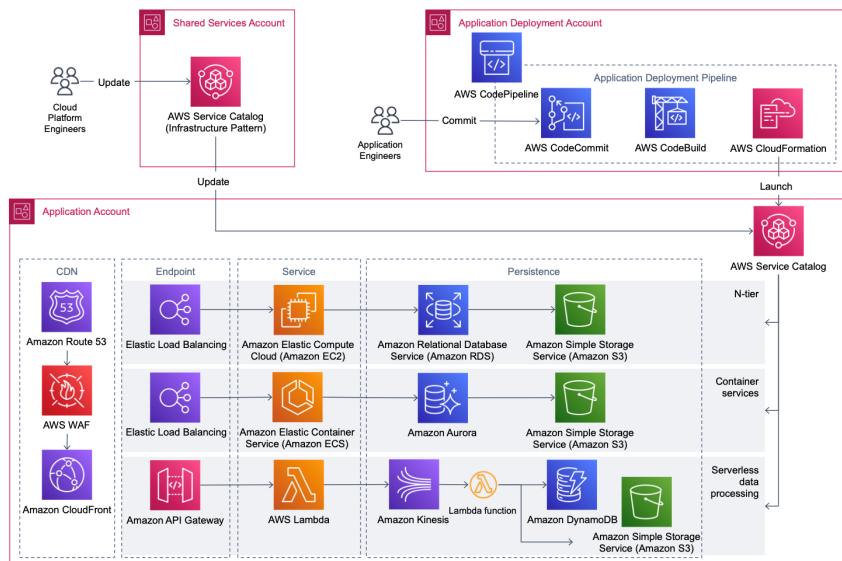


Figure 6.2: Exemple de modèle d'architecture dans votre catalogue de services informatiques

Chapter 7

Documentation

Avant toute chose, rendez-vous sur : <https://docs.aws.amazon.com/>

En-tête de navigation

- **Mise en route** : guides pour débuter (première instance EC2, stockage S3, etc.).
- **Services / Outils / IA** : accès rapide aux familles de documentation par thème technique.
- **Langue, préférences, contact** : paramètres d'affichage et d'assistance.
- **Recherche** : permet de trouver rapidement un service ou un guide précis.
- **Revenir à la console** : lien direct vers la console AWS.

Section “Featured content”

Services essentiels : **EC2, S3, DynamoDB, RDS, Lambda, VPC, SageMaker, Choosing a generative AI service.**

Bloc “Getting started with AWS”

Liens d’initiation pour les apprenants :

- **Set up your AWS environment** : configuration initiale du compte et de la console.
- **Getting Started Resource Center** : centre d’apprentissage interactif.
- **AWS Cloud Security** : bonnes pratiques de sécurité et de conformité.

Bloc inférieur (Guides thématiques)

- **Hands-on Tutorials** : ateliers pratiques guidés.
- **AWS Prescriptive Guidance** : bonnes pratiques architecturales et de déploiement.
- **AWS Architecture** : modèles et diagrammes d’architectures types.
- **AWS Solutions** : cas d’usage réels (e-commerce, data, IoT...).

Colonne latérale droite

Contient le plan de page et des liens rapides (ex. “Developer Resources”) pour naviguer sans tout faire défiler.

Partie “Product Guides & Developer Resources (PGDR)”

Bibliothèque officielle des guides utilisateurs AWS :

- **Colonne de gauche** : catégories (Compute, Database, Networking, Security, etc.).
- **Zone centrale** : description des services, type et lien vers leur documentation complète.
- **Barre de recherche** : permet de retrouver un service parmi plus de 300 produits.

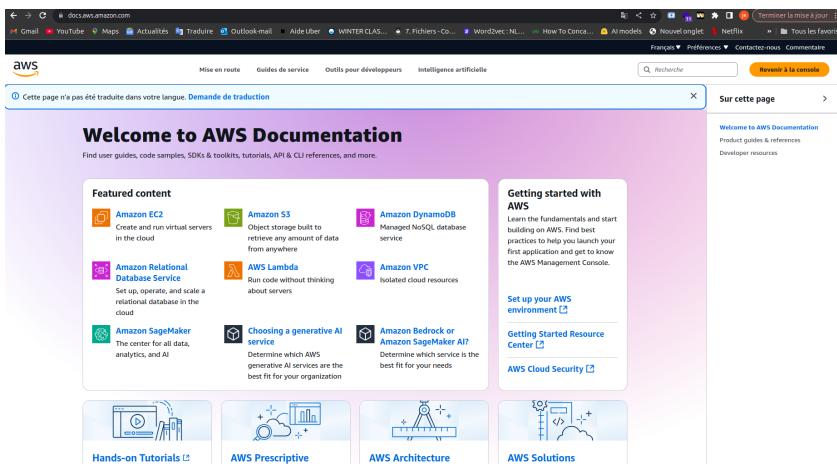


Figure 7.1: Documentation

Product guides & references

Find user guides, developer guides, API references, and CLI references for your AWS products.

Catégorie de produit	Tous les produits (331)
Tous les produits (331)	Rechercher documentation produit
Analytics (23)	Amazon A2I Easily implement human review of ML predictions Machine Learning
Application Integration (9)	Amazon API Gateway Build, deploy, and manage APIs Networking & Content Delivery Serverless
AWS Management Console (4)	Amazon AppFlow No-code integration for SaaS apps and AWS services Analytics
Blockchain (2)	Amazon Application Recovery Controller (ARC) Move traffic for application disaster recovery Networking & Content Delivery
Business Applications (14)	Amazon AppStream 2.0 Stream desktop applications securely to a browser End User Computing
Cloud Financial Management (4)	Amazon Athena Query data in Amazon S3 using SQL Analytics
Compute (17)	Amazon Aurora High performance managed relational database engine Database
Compute HPC (1)	Amazon Aurora DSQL Fapest serverless distributed SQL database for always available applications Database
Containers (6)	Amazon Bedrock Access best-in-class foundation models to build generative AI applications Machine Learning
Cryptography & PKI (7)	
Customer Enablement Services (6)	
Database (12)	
Decision Guides (26)	
Developer Tools (35)	
End User Computing (6)	
Front-End Web & Mobile (9)	
Game Development (3)	
General Reference (7)	

Figure 7.2: PGDR

Chapter 8

Developer Resources

8.1 Developer Resources

Espace dédié aux **développeurs, ingénieurs et formateurs** pour automatiser et intégrer AWS dans le code. On y retrouve les outils principaux suivants :

Outils de développement

- **AWS CLI** : interface en ligne de commande permettant d’interagir directement avec AWS.
- **SDKs & Toolkits** : bibliothèques pour différents langages (*Python (boto3), Java, JavaScript, .NET, Go, Rust, PHP, Swift, etc.*). Utilisées pour automatiser les déploiements, développer des applications ou scripts.
- **AWS CDK (Cloud Development Kit)** : infrastructure as code (IaC) écrite en langage de programmation (*Python, TypeScript, etc.*) au lieu de fichiers JSON ou YAML.

- **Amazon Q Developer & intégrations IDE** : extensions pour *Visual Studio* et *VS Code* afin de travailler directement depuis un environnement de développement.

Ressources complémentaires

- **Code Example Library** : exemples de code concrets utilisant les services AWS.
- **AWS Builder Center** : communauté de projets et ressources collaboratives.
- **Developer Tools Blog** : articles techniques officiels et annonces de nouveautés.
- **AWS re:Post** : forum communautaire d'entraide entre développeurs AWS.

Developer resources

Find AWS software development kits (SDKs) & toolkits, code examples, and more.

AWS CLI
The AWS Command Line Interface (AWS CLI) is a unified tool to manage your AWS products

AWS CLI User Guide

Install the latest version of the AWS CLI

Configure the AWS CLI

SDKs & toolkits

[View all SDKs & toolkits](#)

 AWS CDK	 C++	 Go	 Java	 JS JavaScript
 Kotlin	 .NET	 PHP	 Python	 Ruby
 Rust	 Swift	 SAP ABAP	 Amazon Corretto	 Azure DevOps
 JetBrains	 PowerShell	 Amazon Q Developer	 Visual Studio	 Visual Studio Code

Code Example Library
Find code examples that show you how to use AWS SDKs with AWS

AWS Builder Center
Learn, build, and connect with builders in the AWS community

AWS Developer Tools Blog
Find blog posts for developers written by AWS experts

AWS re:Post
Find expert-reviewed answers, Knowledge Center articles and videos, and curated knowledge paths

Figure 8.1: Ressources pour développeurs

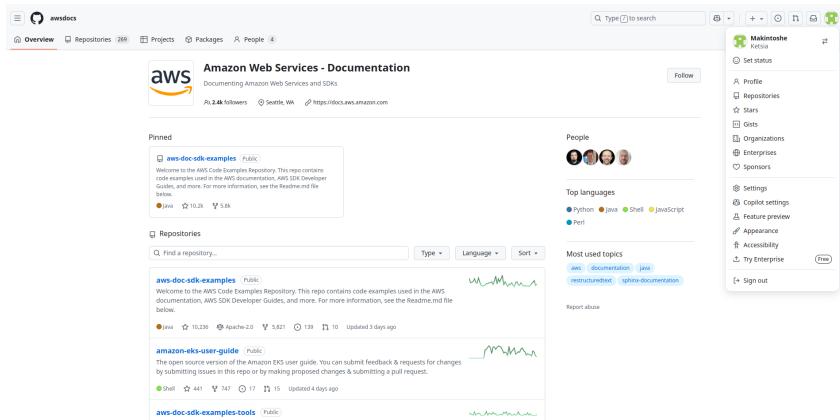


Figure 8.2: Codes sources de la Documentation AWS sur Github

Chapter 9

Atelier

Le mot d'ordre est : pensez archi, maîtrisez les abstractions.

Niveau 1 : IAM ou Organisation et délégation des accès

Créez une structure IAM pour l'équipe Orange Digital, en vous inspirant de l'architecture hiérarchique. (voir l'architectue niveau 1 plus bas). Chaque niveau correspond à un profil IAM différent, mais pour simplifier les manipulations, on peut tout construire à partir du compte “Lead”, qui dispose du rôle AdministratorAccess. Ce compte crée ensuite les autres utilisateurs et leur assigne les autorisations nécessaires. Donc une fois que le Lead est créé, connectez-vous à la console AWS avec le Lead. On verra dans la prochaine partie comment utiliser la CLI.

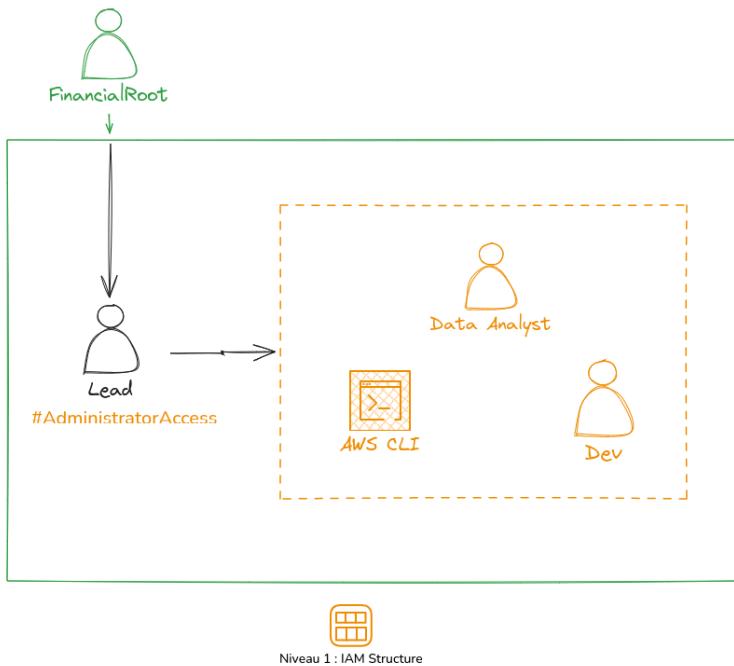


Figure 9.1: Niveau 1 : IAM

Bibliography

[1] Amazone Web Service (AWS):

<http://aws.amazon.com/fr>

[2] Documentation Amazone Web Service (AWS):

<https://docs.aws.amazon.com/>