



Université de Carthage
جامعة قرطاج

SUP'COM

Ecole Supérieure des Communications de Tunis

République Tunisienne

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Carthage
École Supérieure des communications de Tunis

Rapport de stage d'été

Étude et déploiement des services Cloud sur AWS

Département Informatique, Réseaux

Réalisé par

Barkaoui Chaker

Encadré par
Asma BEN LETAIFA

Maitre-Assistante en Télécommunication

Contents

List of Figures	3
List of Tables	4
Remerciement	5
1 Introduction sur AWS et ses services	6
1.1 Introduction	6
1.2 Types et avantages du Cloud Computing AWS	8
1.3 Plateforme cloud Amazon Web Services	11
2 Implémentation des services AWS	20
2.1 Lancement et connexion à une instance AWS EC2	20
2.2 Sécurité du cloud AWS	23
2.3 Développement web sur AWS:	31
2.4 Machine Learning sur AWS:	38
3 Introduction à DevOps sur AWS	44
3.1 Agile Evolution to DevOps	45
3.2 Infrastructure en tant que code	46
3.3 Déploiement continu	48
3.4 Automatisation:	49
3.5 Surveillance	50
3.6 Sécurité	51
Conclusion	53
Bibliographie	54

List of Figures

1.1	Cloud Computing: "Amazon Web Services"	6
1.2	Cloud Computing: un marché en pleine expansion.	7
1.3	Amazon Web Services	7
1.4	Les offres de AWS	9
1.5	Cloud Services Models	10
1.6	AWS Services	11
1.7	Amazon EC2	12
1.8	Amazon Lightsail:	13
1.9	Elastic Load Balancer	15
1.10	Elastic Block Storage	16
1.11	Simple Storage Service	17
1.12	Relational Database Service	17
1.13	Dynamo DB	18
1.14	Amazon CloudWatch	18
1.15	Amazon EC2 Auto Scaling	19
2.1	AWS Console.	20
2.2	Instance Type	21
2.3	Launch Instance	22
2.4	Connected Instance	23
2.5	IAM rôle	24
2.6	JSON example	25
2.7	Liste of Permissions	26
2.8	Amazon Cognito	27
2.9	Integrated firewall	29
2.10	Virtual Private Cloud	29
2.11	AWS Shield	30
2.12	Connect to Github Repository	31
2.13	React Application	32
2.14	AWS Amplify configuration	33
2.15	AWS ECR Permission	39
2.16	ML Instance type	39
2.17	Connect to the instance	40
2.18	AWS ECR Access Verification	40
2.19	Login to AWS ECR	41
2.20	Run AWS DL Containers	41
2.21	Running the DL model	42

2.22	SageMaker Services Architecture	42
2.23	SageMaker Studio	43
3.1	DevOps Démarche	45
3.2	DevOps-Agile	45
3.3	AWS CloudFormation example	47
3.4	AWS CloudFormation template for launching an EC2 instance	47
3.5	AWS CodeDeploy process	49
3.6	AWS OpsWorks showing DevOps features and architecture	50
3.7	Example DevOps automation using Amazon CloudWatch and Auto Scaling	51

Remerciement

Je remercie Madame Asma Ben LETAIFA, enseignante à l'École Supérieure des Communications de Tunis (SUP'COM) et ma maîtresse de stage, pour sa confiance et les connaissances qu'elle a su partager avec moi. Je la remercie aussi pour sa disponibilité et la qualité de son encadrement à SUP'COM, qui m'a fourni les outils nécessaires au bon déroulement de mon stage. Je tiens à la remercier spécialement car elle fut le premier à me soutenir dans ma démarche de stage.

Je saisir cette occasion pour adresser mes profonds remerciements aux responsables et au personnel de l'École Supérieure des Communications de Tunis.

Un grand merci à ma mère et mon père, pour leurs conseils ainsi que leur soutien inconditionnel, à la fois moral et économique.

Chapter 1

Introduction sur AWS et ses services

Dans ce chapitre on va découvrir les services et les connaissances basics sur AWS:



Figure 1.1: Cloud Computing: "Amazon Web Services"

1.1 Introduction

Le cloud computing fait référence à la fourniture à la demande de ressources et d'applications informatiques via Internet.

Avant le cloud computing, nous devions construire des centres de données basés sur la supposition de nos pics maximums, notre client en souffrira. Si nous avons sur-planifié et dépassé notre besoin maximal, nous avons fini par payer pour des ressources dont nous n'avions pas vraiment besoin.

En 2006, Amazon Web Services (AWS) a commencé à offrir des services d'infrastructure informatique aux entreprises sous forme de services Web, désormais communément appelés cloud computing. L'un des principaux avantages du cloud computing est la possibilité de remplacer les dépenses initiales en capital d'infrastructure par de faibles coûts variables qui évoluent avec votre entreprise. Avec le cloud, les entreprises n'ont plus besoin de planifier et de se procurer des serveurs et d'autres infrastructures informatiques des semaines ou des mois à l'avance.

Au lieu de cela, ils peuvent lancer instantanément des centaines ou des milliers de serveurs en quelques minutes et fournir des résultats plus rapidement.

Aujourd'hui, AWS fournit une plate-forme d'infrastructure hautement fiable, évolutive et à faible coût dans le cloud qui alimente des centaines de milliers d'entreprises dans 190 pays à travers le monde.

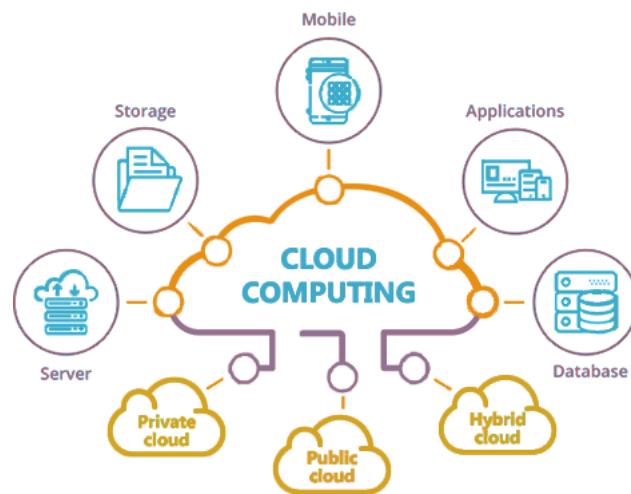


Figure 1.2: Cloud Computing: un marché en pleine expansion.



Figure 1.3: Amazon Web Services

1.2 Types et avantages du Cloud Computing AWS

Six avantages du Cloud Computing

1.Échangez des dépenses en capital contre des dépenses variables - Au lieu de devoir investir massivement dans les centres de données et les serveurs avant de savoir comment vous allez les utiliser, vous ne pouvez payer que lorsque vous consommez des ressources informatiques et ne payez que ce que vous consommez.

2.Bénéficiez d'économies d'échelle massives - En utilisant le cloud computing, vous pouvez obtenir un coût variable inférieur à celui que vous pouvez obtenir seul. Étant donné que l'utilisation de centaines de milliers de clients est agrégée dans le cloud, des fournisseurs tels qu'AWS peuvent réaliser des économies d'échelle plus importantes, ce qui se traduit par des prix à l'utilisation moins élevés.

3.Arrêtez de deviner la capacité - Éliminez les suppositions sur les besoins en capacité de votre infrastructure. Lorsque vous prenez une décision en matière de capacité avant de déployer une application, vous vous retrouvez souvent soit assis sur des ressources inactives coûteuses, soit avec une capacité limitée. Avec le cloud computing, ces problèmes disparaissent. Vous pouvez accéder à autant ou aussi peu de capacité que vous en avez besoin, et augmenter et réduire selon vos besoins avec un préavis de seulement quelques minutes.

4.Augmentez la vitesse et l'agilité - Dans un environnement de cloud computing, les nouvelles ressources informatiques ne sont qu'à un clic, ce qui signifie que vous réduisez le temps de mise à disposition de ces ressources pour vos développeurs de quelques semaines à quelques minutes. Cela se traduit par une augmentation spectaculaire de l'agilité de l'organisation, car le coût et le temps nécessaires pour expérimenter et se développer sont nettement inférieurs.

5.Arrêtez de dépenser de l'argent pour gérer et entretenir les centres de données - Concentrez-vous sur les projets qui différencient votre entreprise, pas l'infrastructure. Le cloud computing vous permet de vous concentrer sur vos propres clients, plutôt que sur la lourde charge de mise en rack, d'empilage et d'alimentation des serveurs.

6.Go global in minutes – Easily deploy your application in multiple regions around the world with just a few clicks. This means you can provide lower latency and a better experience for your customers at minimal cost.



Figure 1.4: Les offres de AWS

Types du Cloud Computing:

Le cloud computing permet aux développeurs et aux services informatiques de se concentrer sur ce qui compte le plus et d'éviter des travaux indifférenciés tels que l'approvisionnement, la maintenance et la planification des capacités. À mesure que le cloud computing gagne en popularité, plusieurs modèles et stratégies de déploiement différents sont apparus pour aider à répondre aux besoins spécifiques de différents utilisateurs. Chaque type de service cloud et de méthode de déploiement vous offre différents niveaux de contrôle, de flexibilité et de gestion. Comprendre les différences entre l'infrastructure en tant que service, la plateforme en tant que service et le logiciel en tant que service, ainsi que les stratégies de déploiement que vous pouvez utiliser, peut vous aider à décider quel ensemble de services est adapté à vos besoins.

Modèles de cloud computing:

Infrastructure as a Service (IaaS):

Infrastructure as a Service (IaaS) contains the basic building blocks for cloud IT and typically provides access to networking features, computers (virtual or on dedicated hardware), and data storage space. IaaS provides you with the highest level of flexibility and management control over your IT resources and is most similar to existing IT resources that many IT departments and developers are familiar with today.

Platform as a Service (PaaS):

Platform as a Service (PaaS) removes the need for your organization to manage the underlying infrastructure (usually hardware and operating systems) and allows you to focus on the deployment and management of your applications. This helps you be more efficient as you don't need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application.

Software as a Service (SaaS):

Software as a Service (SaaS) provides you with a completed product that is run and managed by the service provider. In most cases, people referring to Software as a Service are referring to end-user applications. With a SaaS offering you do not have to think about how the service is maintained or how the underlying infrastructure is managed; you only need to think about how you will use that particular piece of software. A common example of a SaaS application is web-based email which you can use to send and receive email without having to manage feature additions to the email product or maintain the servers and operating systems that the email program is running on.

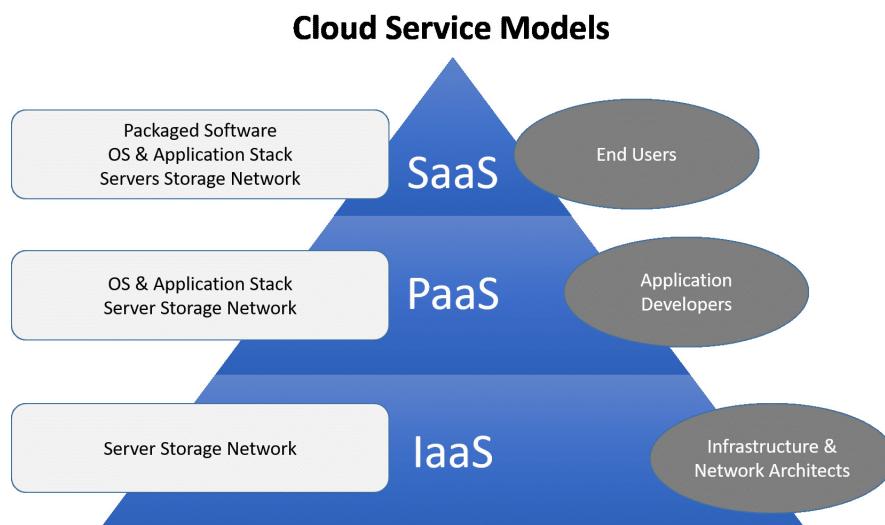


Figure 1.5: Cloud Services Models

1.3 Plateforme cloud Amazon Web Services

Dans cette partie, on se propose d'étudier les différents services de AWS.

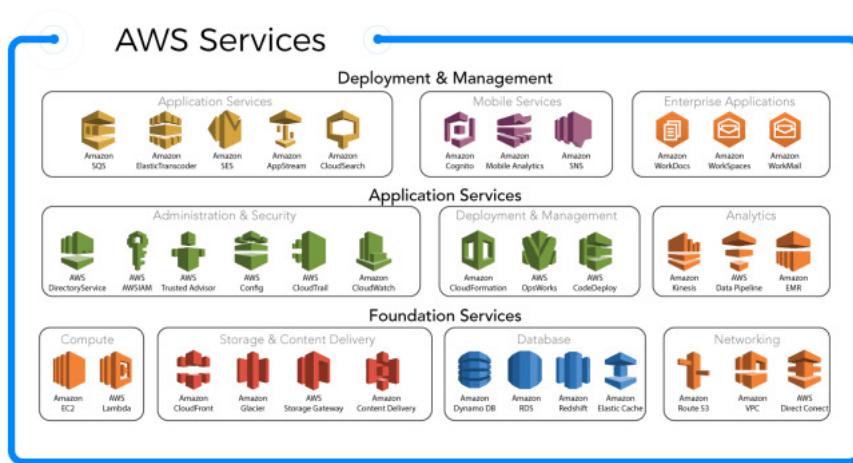


Figure 1.6: AWS Services

L'infrastructure cloud AWS est construite autour de régions et de zones de disponibilité. Une région est une zone géographiquement séparée qui n'est pas en réalité un seul emplacement, mais une combinaison de plusieurs zones de disponibilité. Chaque zone de disponibilité peut en fait avoir plusieurs centres de données ou emplacements, mais tous sont logiquement traités comme une seule unité.

La raison d'avoir plusieurs zones de disponibilité dans une région est la réPLICATION et éviter un point de défaillance unique des services. Même s'il y a une circonstance imprévue qui conduit à un échec de la connectivité à l'une des zones de disponibilité, une copie de secours de toutes les données et applications existe sur une autre zone de disponibilité dans la même région et ainsi vos clients peuvent toujours accéder à vos applications et services parfaitement.

Les zones de disponibilité peuvent cependant être isolées et distantes de plusieurs kilomètres sont connectées avec une latence très faible et un débit élevé à l'aide de plusieurs réseaux fibre optique haut débit propriétaires.

Choses à garder à l'esprit lors du choix d'une région spécifique.

*Coût - Différents coûts basés sur les économies régionales.

*Latence - À mesure que la distance augmente entre vos serveurs d'applications et vos clients, la latence augmente.

*Conformité - Différents pays et régions ont des lois différentes.

*Disponibilité des services - Tous les services peuvent ne pas être disponibles dans toutes les régions.

****Compute Services:**

Pour exécuter vos applications dans le cloud, vous avez besoin d'une puissance de calcul disponible 24h / 24 et 7j / 7. Dans un environnement traditionnel, vous devez entretenir physiquement les serveurs ainsi que les logiciels qui fonctionnent dessus et vous avez également besoin d'une très bonne estimation de la quantité de services informatiques dont vous avez besoin, en revanche AWS fournit des services informatiques à la demande. , plus flexible en termes de capacité et ne nécessitant aucun entretien physique.

1. Amazon Elastic Compute Cloud (EC2)
2. Amazon Lightsail
3. AWS Lambda
4. Amazon Elastic Container Services (ECS)
5. Amazon Elastic Container Service for Kubernetes (EKS)



Figure 1.7: Amazon EC2

Amazon EC2:

Amazon Elastic Compute Cloud (Amazon EC2) est un service Web qui fournit une capacité de calcul sécurisée et redimensionnable dans le cloud. Il exécute un serveur virtuel et présente un environnement informatique virtuel avec une variété de systèmes d'exploitation et de configurations matérielles disponibles sous la forme de différentes AMI ou Amazon Machine Images. Les instances EC2 sont disponibles dans différentes tailles en ce qui concerne la mémoire et le nombre de processeurs. En outre, il existe différents types d'instances EC2 en fonction des cas d'utilisation distincts pour lesquels elles sont optimisées.

Amazon EC2 vous permet de choisir entre des instances de performances fixes (par exemple, M5, C5 et R5) et des instances de performances extensibles (par exemple T3). Les instances de performances extensibles fournissent un niveau de base de performances du processeur avec la possibilité d'éclater au-dessus de la ligne de base.

Amazon EC2 vous permet de provisionner une variété de types d'instances, qui fournissent différentes combinaisons de CPU, mémoire, disque et réseau. C'est une bonne idée d'exécuter des tests sur plusieurs instances avec différentes combinaisons de ressources pour déterminer celle qui fonctionne le mieux.



Figure 1.8: Amazon Lightsail:

Amazon Lightsail:

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

Lightsail fournit aux développeurs des capacités de calcul, de stockage et de mise en réseau, et il fournit également des capacités pour déployer et gérer des sites Web et des applications Web dans le cloud, tout comme EC2, mais il en est une version plus simple et simplifiée.

****Services de réseau:**

Virtual Private Cloud:

Un cloud privé virtuel (VPC) sur AWS est un réseau virtuel qui isole logiquement toutes vos instances et applications du reste du cloud AWS. Il fournit un cadre ou une boîte virtuelle dans laquelle toutes vos instances vivent à l'intérieur, et l'idée est que rien n'entre dans la boîte, rien ne sort de la boîte sans votre autorisation spécifique, donc cela vous permet de maintenir et d'avoir un contrôle complet sur tous les actifs à l'intérieur du VPC.

Il sert principalement les objectifs suivants:

Isoler vos ressources AWS des autres comptes.

Acheminer le trafic réseau vers et depuis vos instances.

Protéger vos instances contre les intrusions sur le réseau.

Lorsque vous créez un VPC, vous divisez également l'espace à l'intérieur du VPC en créant des sous-réseaux et nous lançons les instances EC2 à l'intérieur de ces sous-réseaux. Les sous-réseaux sont principalement utilisés pour déterminer l'accès aux passerelles, pour entrer / sortir et pour isoler le trafic spécifique dont vous ne voulez pas vous parler ou vous parler.

Pour configurer un VPC, vous n'avez besoin que de deux choses:

1. Région
2. Plage d'adresses IP - pour les adresses IP privées de toutes vos ressources à l'intérieur du VPC

Vous pouvez créer un VPC en accédant aux services et en recherchant un VPC, mais lorsque vous lancez une instance EC2, AWS crée pour vous un VPC par défaut que vous pouvez simplement personnaliser selon vos besoins.

Mais pour le moment, toutes nos ressources existent dans une seule zone de disponibilité, pour aller au-delà, nous devons créer des sous-réseaux supplémentaires dans une zone de disponibilité différente. Chaque fois que vous créez un nouveau sous-réseau public, vous devez l'associer à votre table de routage pour autoriser le trafic public.

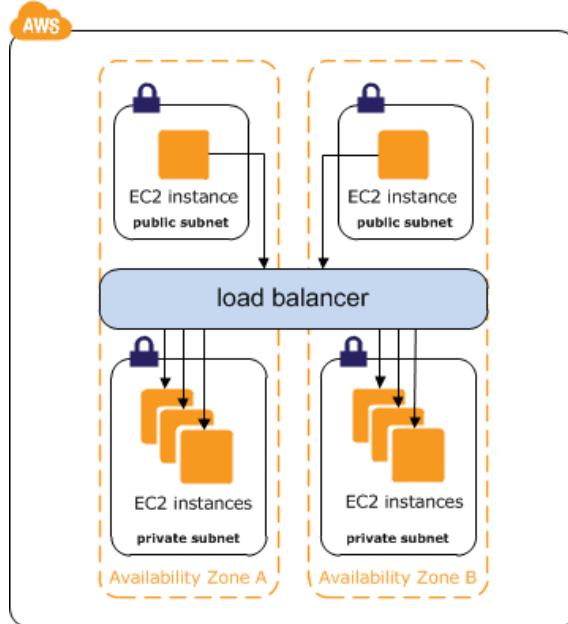


Figure 1.9: Elastic Load Balancer

Elastic Load Balancer (ELB):

Lorsque vous faites cela, vous disposez de deux adresses IP publiques via lesquelles vous pouvez accéder à votre application. Nous devons donc créer un Elastic Load Balancer (ELB) et l'associer à nos deux instances afin que le trafic soit automatiquement réparti entre les deux.

ELB propose trois types d'équilibriseurs de charge qui offrent tous la haute disponibilité, la mise à l'échelle automatique et la sécurité robuste qui sont nécessaires pour rendre les applications tolérantes aux pannes.

Un Equilibreur de charge d'application fonctionne au niveau de la demande (couche 7), acheminant le trafic vers des cibles - telles que des instances EC2, des microservices et des conteneurs - au sein d'Amazon VPC, en fonction du contenu de la demande. Il est idéal pour l'équilibrage de charge avancé du trafic HTTP (Hypertext Transfer Protocol) et HTTP sécurisé (HTTPS).

Un équilibrleur de charge réseau fonctionne au niveau de la connexion (couche 4), acheminant les connexions vers des cibles - telles que les instances Amazon EC2, les microservices et les conteneurs - dans Amazon VPC, en fonction des données de protocole IP. Il est idéal pour l'équilibrage de charge du trafic TCP (Transmission Control Protocol).

L'équilibrleur de charge classique fournit un équilibrage de charge de base sur plusieurs instances Amazon EC2 et fonctionne à la fois au niveau de la demande et au niveau de la connexion.

****Services de stockage:**

Les options de stockage AWS permettent aux clients de stocker et d'accéder à leurs données sur Internet de manière durable, fiable et économique. Il existe un certain nombre d'options de stockage disponibles qui sont mieux utilisées à des fins différentes.

Amazon S3 qui a un stockage au niveau objet et Amazon RDS qui s'exécute sur un stockage au niveau bloc. Le stockage pour les bases de données et EC2 utilise le stockage au niveau des blocs comme Amazon Elastic Block Storage (EBS)

What is Amazon EBS?

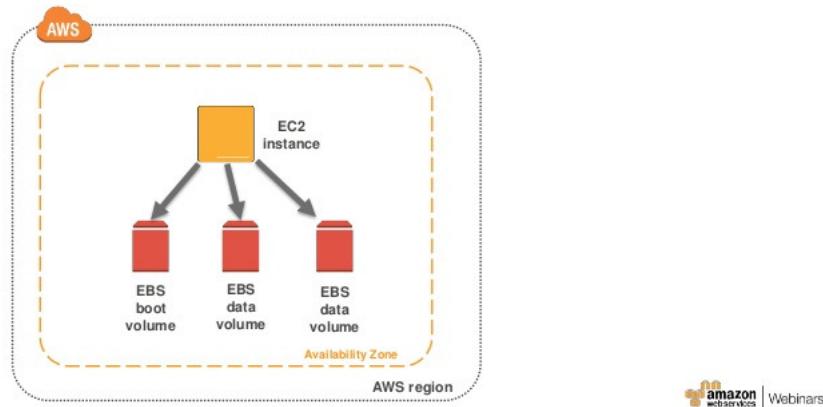


Figure 1.10: Elastic Block Storage

Elastic Block Storage:

EBS fournit le stockage par blocs le plus courant disponible sur le cloud AWS. Lorsque vous lancez une instance EC2, vous avez besoin d'un stockage en bloc pour l'accompagner, de sorte qu'un volume de démarrage et un volume de données se connectent directement à l'instance EC2. Ces blocs de stockage sont stockés indépendamment et existent donc même si les instances EC2 sont arrêtées. Vous pouvez simplement prendre ces blocs de stockage et les attacher à différentes instances.

Chaque volume Amazon EBS est automatiquement répliqué dans une zone de disponibilité pour vous protéger des pannes de composants, ce qui offre une haute disponibilité et une durabilité.

Simple Storage Service:

Amazon Simple Storage Service, ou S3, est une solution de stockage d'objets hautement évolutive et durable, pour stocker et récupérer n'importe quelle quantité de données. S3 est un moyen simple et économique de stocker et de récupérer des données à tout moment, de n'importe où sur le Web. S3 est couramment utilisé pour le stockage d'objets de base comme des images ou des fichiers texte, le stockage de sauvegardes, l'hébergement de fichiers d'application, l'hébergement multimédia, etc.



Figure 1.11: Simple Storage Service

Trois copies de données sont stockées de manière redondante dans les installations AWS d'une région pour assurer un haut niveau de durabilité, par conséquent S3 peut être utilisé comme un endroit pour stocker confortablement les sauvegardes.

****Bases de données sur AWS:**



Figure 1.12: Relational Database Service

Relational Database Service:

Un moyen simple de configurer, d'exploiter et de mettre à l'échelle une base de données relationnelle dans le cloud. Il offre une capacité rentable et redimensionnable tout en automatisant les tâches d'administration chronophages telles que l'approvisionnement du matériel, la configuration de la base de données, les correctifs et la réalisation de sauvegardes.

Amazon RDS prend actuellement en charge six moteurs de base de données:

Amazon Aurora: <https://aws.amazon.com/rds/aurora/>

PostgreSQL: <https://aws.amazon.com/rds/postgresql/>

MySQL: <https://aws.amazon.com/rds/mysql/>

MariaDB: <https://aws.amazon.com/rds/mariadb/>

Oracle: <https://aws.amazon.com/rds/oracle/>

Microsoft SQL Server: <https://aws.amazon.com/rds/sqlserver/>



Figure 1.13: Dynamo DB

Dynamo DB::

DynamoDB est une solution de base de données NoSQL entièrement gérée, rapide et évolutive qui offre des performances fiables à n'importe quelle échelle. Puisqu'il s'agit de NoSQL, il s'agit d'une base de données sans schéma et ne nécessite qu'un nom de table et une clé primaire pour créer une table Dynamo DB.

****Surveillance:**



Figure 1.14: Amazon CloudWatch

Amazon CloudWatch:

Amazon CloudWatch est un service de surveillance des ressources AWS Cloud et des applications que vous exécutez sur AWS. Il collecte des points de données au fil du temps et fournit des statistiques sur l'infrastructure. Ainsi, nous pouvons ensuite créer des alarmes et déclencher des événements automatisés basés sur ces statistiques. Nous pouvons automatiquement mettre à l'échelle les systèmes avant qu'ils ne commencent à affecter les utilisateurs. Il peut être utilisé pour définir des alarmes, visualiser les journaux et les mesures côté à côté, prendre des mesures automatisées pour la mise à l'échelle, résoudre les problèmes opérationnels et découvrir des informations pour optimiser les applications.

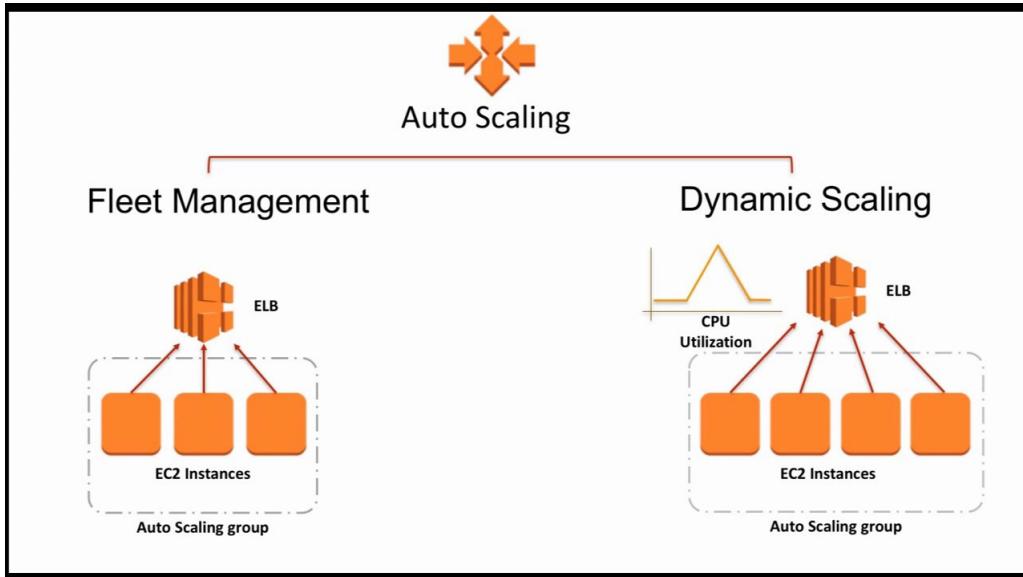


Figure 1.15: Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling:

contribue à maintenir la disponibilité des applications, en permettant une mise à l'échelle dynamique de la capacité Amazon EC2 à la hausse ou à la baisse automatiquement selon des conditions prédéfinies. Il peut être utilisé pour mettre à l'échelle dynamiquement des instances Amazon EC2. La mise à l'échelle dynamique augmente automatiquement le nombre d'instances Amazon EC2 pendant les pics de demande pour maintenir les performances et réduire la capacité pendant les accalmies, ce qui peut aider à réduire les coûts. Amazon EC2 Auto Scaling est bien adapté aux applications qui ont des modèles de demande stables ou aux applications qui connaissent une variabilité d'utilisation horaire, quotidienne ou hebdomadaire.

Cela couvre à peu près les bases d'Amazon Web Services. Le seul problème que j'ai avec le cours est qu'il aurait pu être un peu plus pratique. Ils prévoient de lancer des cours de suivi plus spécifiques et plus détaillés sur les différents services proposés, alors restez à l'écoute pour en savoir plus.

Chapter 2

Implémentation des services AWS

2.1 Lancement et connexion à une instance AWS EC2

1. Rendez-vous sur <https://aws.amazon.com/> et connection à AWS Console.

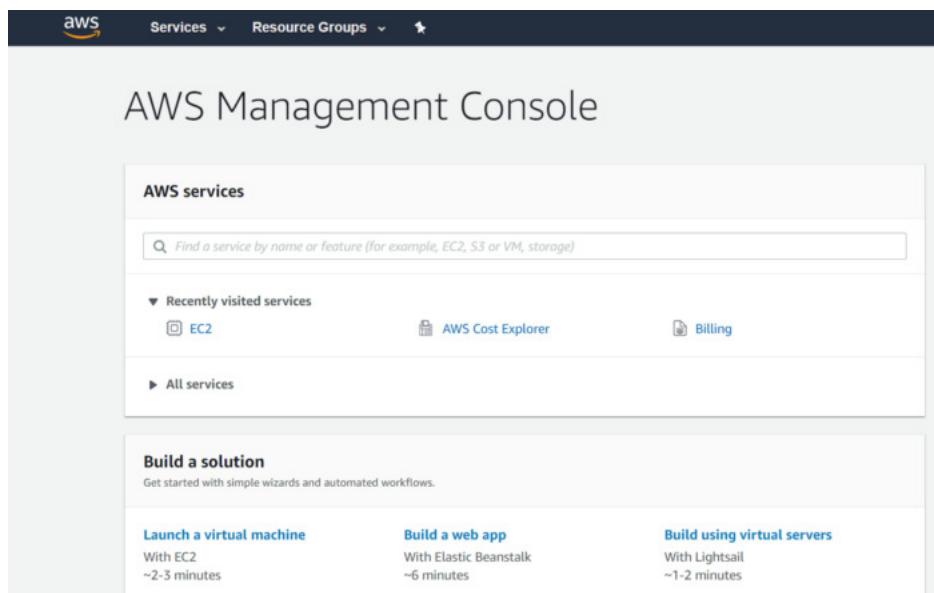


Figure 2.1: AWS Console.

2. Cliquez maintenant sur services, puis sélectionnez EC2. Cliquez ensuite sur le bouton Lancer l'instance.
3. Ensuite, vous devez choisir une AMI (Amazon Machine Image), qui contient la configuration logicielle, c'est-à-dire le système d'exploitation, le serveur d'applications et les applications nécessaires pour lancer l'instance.

Dans ce rapport, j'utiliserai l'image Amazon Linux AMI 2018.03.0. L'AMI Amazon Linux est construite et maintenue par Amazon lui-même et offre donc la meilleure intégration et est également pré-installée avec de nombreux outils utiles.

4. Sélectionnez le type d'instance, t2.micro est la seule instance disponible pour l'offre gratuite et vous offre une mémoire de 1 Go. Appuyez sur Review and Launch, puis lancez à nouveau sur la page suivante que vous voyez ci-dessous.

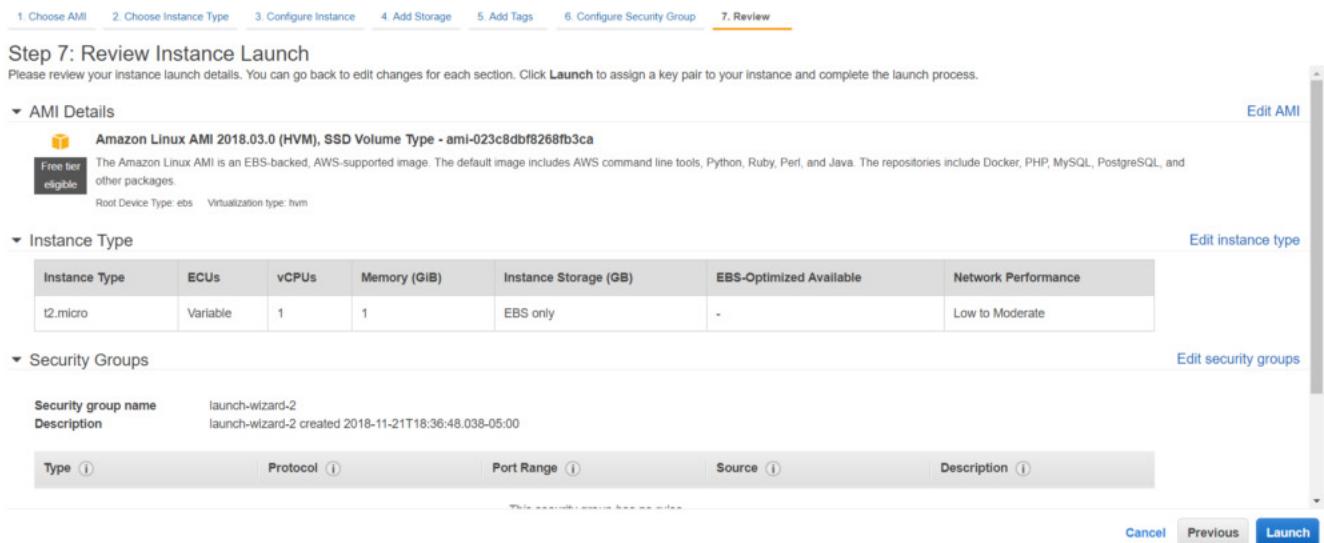


Figure 2.2: Instance Type

5. Ensuite, vous devez créer une paire de clés qui vous permettra de vous connecter à votre instance en toute sécurité. Sélectionnez «Créer une nouvelle paire de clés», donnez-lui un nom puis assurez-vous de télécharger la paire de clés, cela vous donnera un fichier «.pem», vous en aurez besoin pour vous connecter à votre instance plus tard. Après avoir téléchargé la paire de clés, lancez l'instance.

6. Sur la page suivante, faites défiler vers le bas et cliquez sur «Afficher les instances». L'initialisation peut prendre un peu de temps. Une fois l'initialisation terminée, votre instance doit être opérationnelle.

Connexion à l'instance EC2.

7. Installez Putty sur votre machine locale. Avant de nous connecter à l'instance, nous devons convertir la paire de clés du fichier «.pem» en un fichier «.ppk» afin que Putty puisse la comprendre.

8. Ouvrez Puttygen qui devrait être pré-installé lorsque vous installez Putty. Si vous ne pouvez pas le trouver dans la recherche Windows (cela m'est arrivé), parcourez la liste de tous les programmes, et il devrait être là dans le dossier nommé Putty.

9. Cliquez sur Charger et sélectionnez la paire de clés, le fichier «.pem», appuyez sur OK puis cliquez sur «Enregistrer la clé privée» et enregistrez-la en tant que fichier «.ppk».

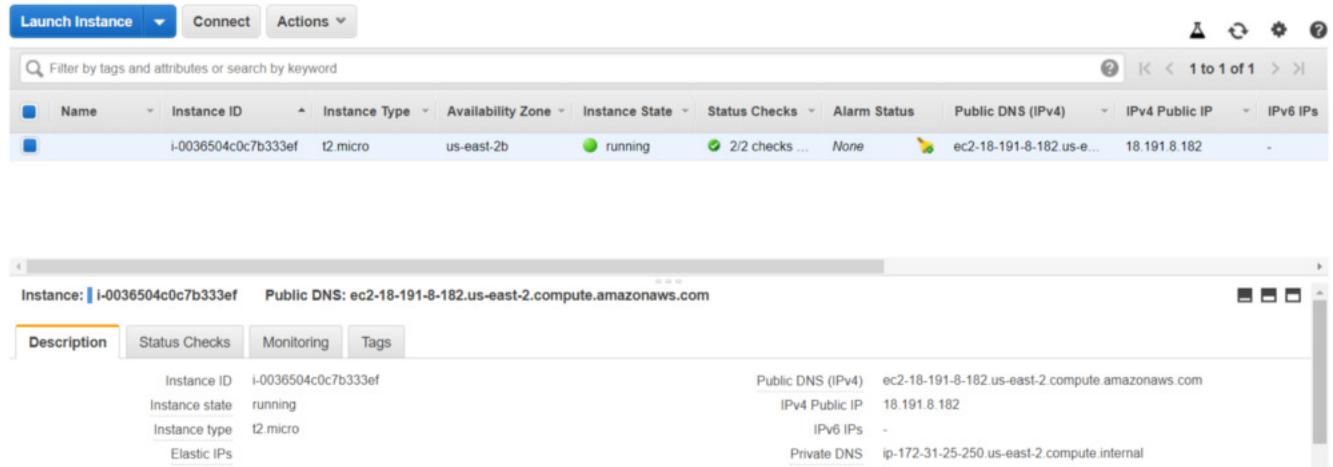


Figure 2.3: LAunch Instance

10. Maintenant, ouvrez Putty. Pour nous connecter à notre instance EC2, nous avons besoin de son nom d'hôte. Vous pouvez le trouver sur la page Instances sur AWS.
11. Ensuite, sous le volet «Catégorie» sur le côté gauche, développez «SSH», puis sélectionnez «Auth». Sous «Paramètres d'authentification», recherchez le fichier «.ppk» que vous avez créé à l'étape 3 et sélectionnez-le. Revenez maintenant à «Session» dans le volet «Catégorie» sur la gauche.
12. Vous pouvez enregistrer cette configuration de session afin de ne pas avoir à tout taper à nouveau la prochaine fois que vous vous connectez. Alors, tapez un nom sous «Sessions enregistrées», puis cliquez sur «Enregistrer». La prochaine fois que vous voudrez le lancer, sélectionnez simplement la session et cliquez sur «Charger», il devrait également charger automatiquement le fichier «.ppk» pour l'authentification.
13. Cliquez sur Ouvrir. Le terminal Putty devrait s'ouvrir et vous recevrez également un avertissement de sécurité. Vous n'avez pas à vous en préoccuper, cliquez simplement sur «Oui».

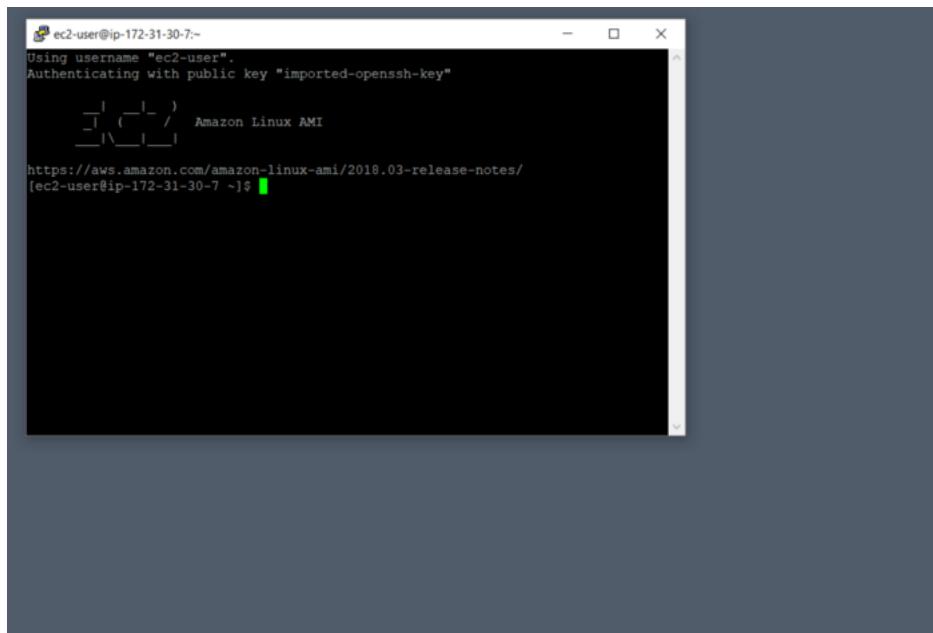


Figure 2.4: Connected Instance

2.2 Sécurité du cloud AWS

1.Gestion et surveillance de l'accès sur AWS:

1.1 Gestion des identifications et des accès(IAM):

La sécurité du cloud est la priorité la plus élevée dans AWS. Lorsque vous hébergez votre environnement dans le cloud, vous pouvez être assuré qu'il est hébergé dans un centre de données ou dans une architecture réseau conçue pour répondre aux exigences de l'organisation la plus sensible à la sécurité. De plus, ce haut niveau de sécurité est disponible sur une base de paiement à l'utilisation, ce qui signifie qu'il n'y a vraiment aucun coût initial, et le coût d'utilisation du service est beaucoup moins cher par rapport à un environnement sur site.

Il existe de nombreux types de services de sécurité disponibles, mais certains d'entre eux sont largement utilisés par AWS, tels que:

*IAM

*Key Management System (KMS)

*Cognito

*Web Access Firewall (WAF)

AWS Identity and Access Management (IAM) est un service Web permettant de contrôler en toute sécurité l'accès aux ressources AWS. Il vous permet de créer et de contrôler des services pour l'authentification des utilisateurs ou de limiter l'accès à un certain ensemble de personnes qui utilisent vos ressources AWS.

Le flux de travail IAM comprend les six éléments suivants:

1.Un principal est une entité qui peut effectuer des actions sur une ressource AWS. Un utilisateur, un rôle ou une application peut être un mandant.

2.L'authentification est le processus de confirmation de l'identité du mandant essayant d'accéder à un produit AWS. Le mandant doit fournir ses informations d'identification ou les clés requises pour l'authentification.

3.Demande: un principal envoie une demande à AWS spécifiant l'action et la ressource qui doit l'exécuter.

4.Autorisation: par défaut, toutes les ressources sont refusées. IAM n'autorise une demande que si toutes les parties de la demande sont autorisées par une stratégie correspondante. Après avoir authentifié et autorisé la demande, AWS approuve l'action.

5.Les actions sont utilisées pour afficher, créer, modifier ou supprimer une ressource.

6.Ressources: un ensemble d'actions peut être effectué sur une ressource liée à votre compte AWS.

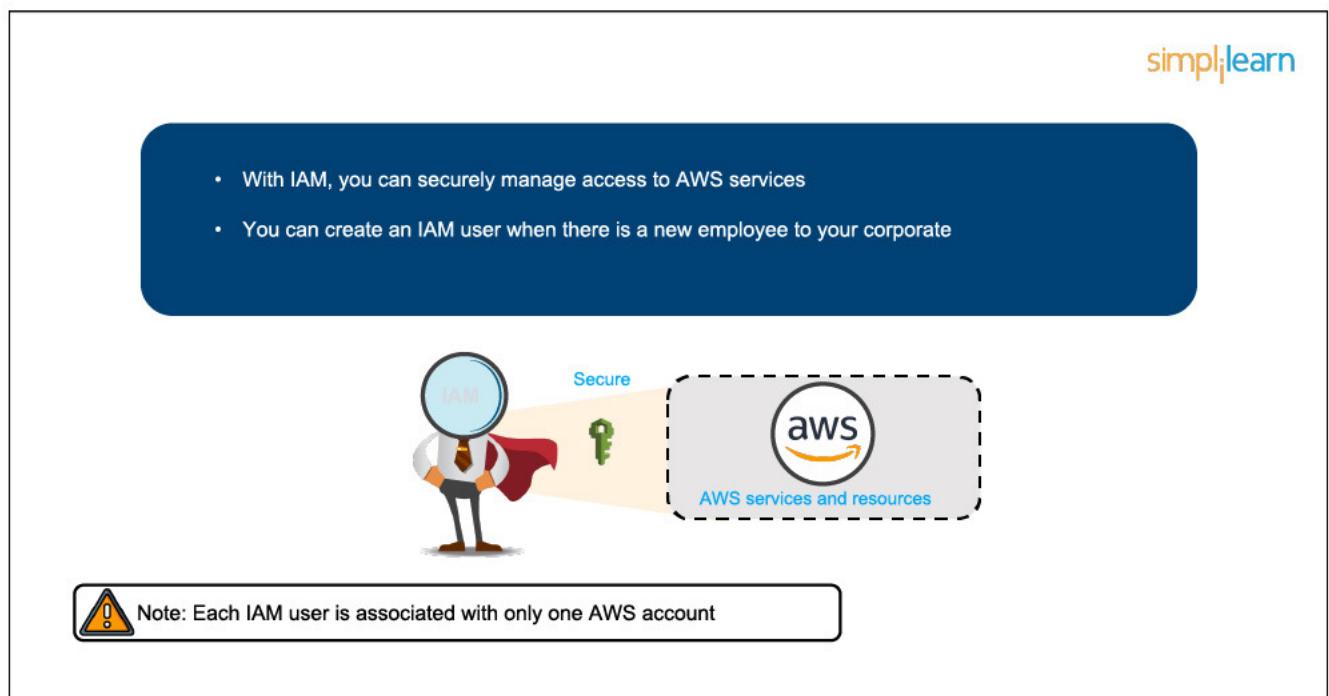


Figure 2.5: IAM rôle

Il existe d'autres composants de base d'IAM. Premièrement, nous avons l'utilisateur; de nombreux utilisateurs forment ensemble un groupe. Les stratégies sont les moteurs qui autorisent ou refusent une connexion en fonction de la stratégie. Les rôles sont des informations d'identification temporaires qui peuvent être utilisées pour une instance selon les besoins.

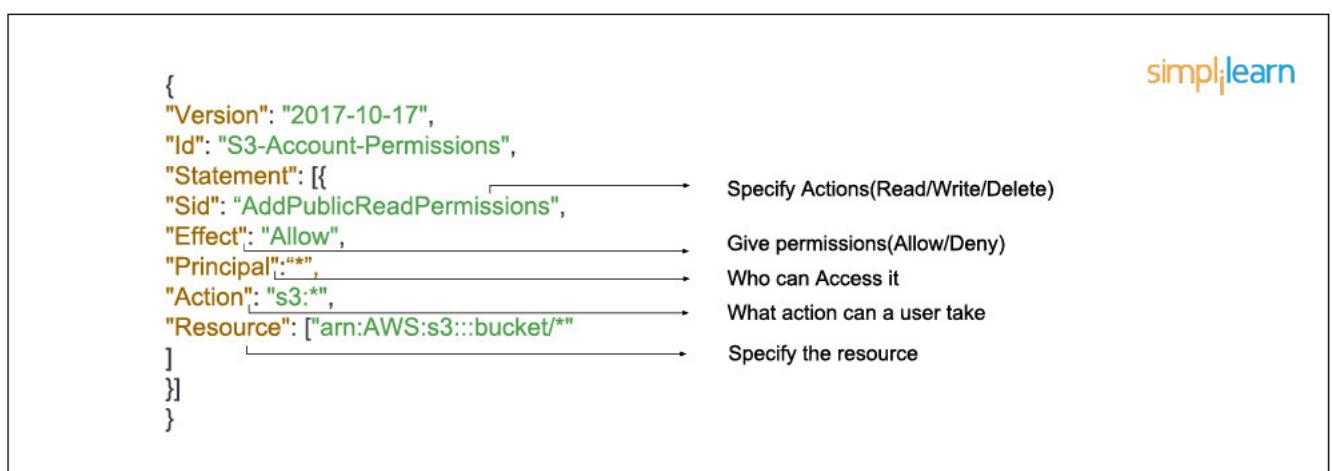
Un utilisateur IAM est une identité à laquelle sont associées des informations d'identification et des autorisations associées. Cela peut être une personne réelle qui est un utilisateur ou une application qui est un utilisateur. Avec IAM, vous pouvez gérer en toute sécurité l'accès aux services AWS en créant un nom d'utilisateur IAM pour chaque employé de votre organisation.

Une collection d'utilisateurs IAM est un groupe IAM. Vous pouvez utiliser des groupes IAM pour spécifier des autorisations pour plusieurs utilisateurs afin que toutes les autorisations appliquées au groupe soient également appliquées aux utilisateurs individuels de ce groupe. La gestion des groupes est assez simple. Vous définissez des autorisations pour le groupe et ces autorisations sont automatiquement appliquées à tous les utilisateurs du groupe.

Une stratégie IAM définit l'autorisation et contrôle l'accès aux ressources AWS. Les stratégies sont stockées dans AWS en tant que documents JSON. Les autorisations spécifient qui a accès aux ressources et quelles actions ils peuvent effectuer. Par exemple, une stratégie peut permettre à un utilisateur IAM d'accéder à l'un des compartiments d'Amazon S3. La politique contiendrait les informations suivantes:

- 1.Qui peut y accéder?
- 2.Quelles actions cet utilisateur peut-il entreprendre?
- 3.Quelles ressources AWS auxquelles l'utilisateur peut accéder?
- 4.Quand ils sont accessibles?

Au format JSON, cela ressemblerait à ceci:



```
{
  "Version": "2017-10-17",
  "Id": "S3-Account-Permissions",
  "Statement": [
    {
      "Sid": "AddPublicReadPermissions", → Specify Actions(Read/Write/Delete)
      "Effect": "Allow", → Give permissions(Allow/Deny)
      "Principal": "*", → Who can Access it
      "Action": "s3:*", → What action can a user take
      "Resource": ["arn:AWS:s3:::bucket/*"] → Specify the resource
    }
  ]
}
```

The screenshot shows a JSON policy document with annotations pointing to specific fields and their meanings. The annotations are as follows:

- Specify Actions(Read/Write/Delete)**: Points to the `Effect` field.
- Give permissions(Allow/Deny)**: Points to the `Effect` field.
- Who can Access it**: Points to the `Principal` field.
- What action can a user take**: Points to the `Action` field.
- Specify the resource**: Points to the `Resource` field.

Figure 2.6: JSON example

Un rôle IAM est un ensemble d'autorisations qui définissent les actions autorisées et refusées par une entité dans la console AWS.

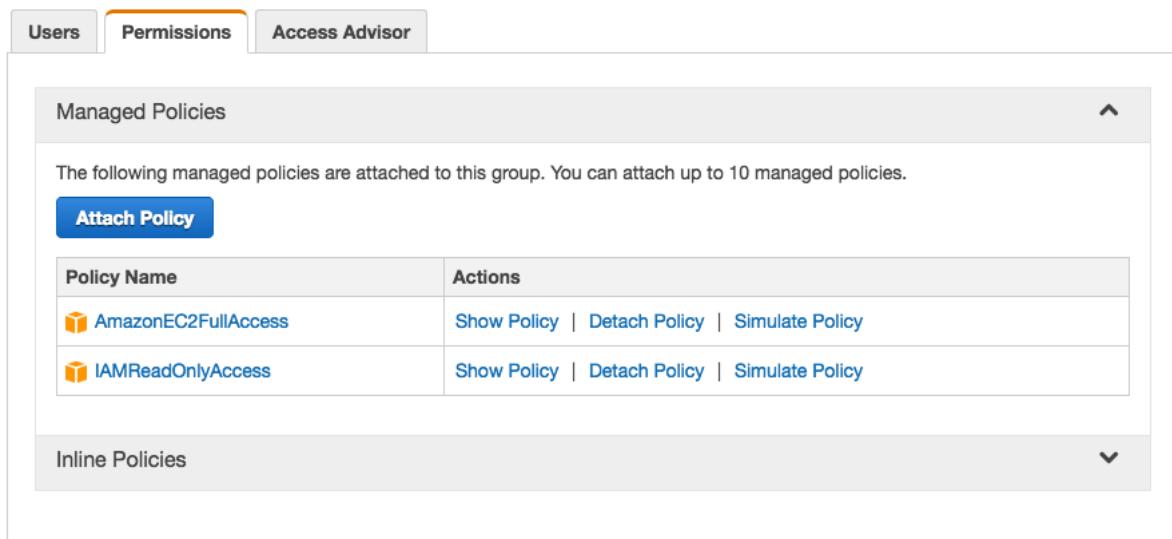


Figure 2.7: Liste of Permissions

1.2 Authentification multifacteur (MFA):

MFA ajoute une sécurité supplémentaire car elle oblige les utilisateurs à fournir une authentification unique à partir d'un mécanisme MFA pris en charge par AWS en plus de leurs informations d'identification de connexion habituelles lorsqu'ils accèdent aux sites Web ou aux services AWS:

1.3 Amazon Cognito:

Amazon Cognito est un service d'annuaire AWS fourni par amazon pour un développement simple et rapide d'applications Web / mobiles. Ce service vous aide à gérer vos fonctions d'authentification, d'autorisation et de gestion des utilisateurs afin que vous puissiez vous concentrer sur la gestion de vos applications plutôt que sur la gestion des utilisateurs et de l'authentification. Le service Cognito propose une connexion / inscription à échelle automatique à l'aide de votre propre groupe d'utilisateurs et une intégration facile avec des fournisseurs d'identité sociale tels que Google, Facebook, Amazon ou vous pouvez intégrer votre propre fournisseur d'identité à l'aide de SAML 2.0.

Cognito vous propose de vous connecter et de vous inscrire en tant que service de plateforme afin que vous puissiez vous concentrer davantage sur la création des fonctionnalités de votre application. Voici quelques-unes des fonctionnalités

Un pool d'utilisateurs Amazon Cognito et un pool d'identités utilisés ensemble

Consultez le diagramme pour un scénario Amazon Cognito typique. Ici, l'objectif est de vérifier votre client et d'attribuer ensuite à votre client l'accès à un autre service AWS.

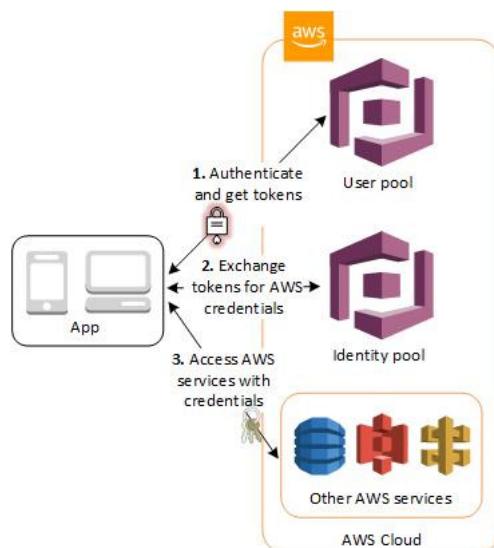


Figure 2.8: Amazon Cognito

Lors de la première étape, l'utilisateur de votre application se connecte via un pool d'utilisateurs et obtient des jetons de pool d'utilisateurs après une vérification réussie.

Ensuite, votre application échange les jetons du pool d'utilisateurs contre des informations d'identification AWS via un pool d'identités.

Enfin, votre client d'application pourrait alors utiliser ces informations d'identification AWS pour accéder à différents services AWS, par exemple, Amazon S3 ou DynamoDB.

2. Sécurisation des données et des secrets sur AWS:**2.1 Capacités de cryptage:**

Tous les services AWS qui gèrent les données client chiffrent les données en mouvement et fournissent des options pour chiffrer les données au repos. Tous les services AWS qui offrent un chiffrement au repos à l'aide d'AWS KMS ou d'AWS CloudHSM utilisent AES-256. Aucun de ces services ne stocke les clés de chiffrement en texte brut au repos - c'est une fonction que seuls AWS KMS et AWS CloudHSM peuvent exécuter à l'aide de leurs HSM validés FIPS 140-2. Cette architecture permet de minimiser l'utilisation non autorisée des clés.

Lors du chiffrement de données en mouvement, les services AWS utilisent le protocole TLS (Transport Layer Security) pour fournir un chiffrement entre votre application et le service AWS. La plupart des solutions commerciales utilisent un projet open source appelé OpenSSL pour leurs besoins TLS.

2.2 Options de gestion des clés (AWS Key Management Service):

AWS Key Management Service (KMS) vous offre un contrôle centralisé des clés de chiffrement utilisées pour protéger vos données. Le service est intégré aux autres services AWS, ce qui simplifie le chiffrement des données que vous stockez dans ces services et le contrôle d'accès aux clés pour les déchiffrer. AWS KMS est intégré à AWS CloudTrail, vous permettant de vérifier qui a utilisé quelles clés, sur quelles ressources et quand. AWS KMS permet aux développeurs d'ajouter facilement une fonctionnalité de signature numérique ou de chiffrement au code de leur application, que ce soit directement ou en utilisant l'AWS SDK. Le SDK de chiffrement AWS prend en charge AWS KMS comme fournisseur de clé principale pour les développeurs qui ont besoin de chiffrer / déchiffrer des données locales dans leurs applications.

3. Sécurité réseau avancée sur AWS:**3.1 Pare-feu intégrés:**

AWS Firewall Manager est un outil de gestion de la sécurité permettant de configurer et de gérer de manière centralisée les règles AWS WAF sur vos comptes et applications. À l'aide du gestionnaire de pare-feu, vous pouvez déployer des règles WAF en même temps pour vos équilibreurs de charge d'application et vos distributions Amazon CloudFront et vous assurer également que les nouvelles applications et ressources sont conformes à un ensemble commun de règles de sécurité dès le premier jour.

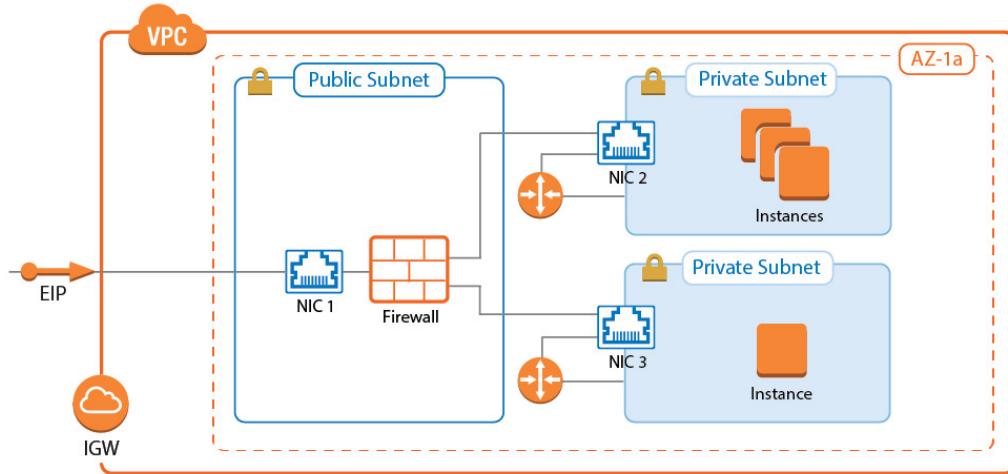


Figure 2.9: Integrated firewall

3.2 Amazon Virtual Private Cloud:

Amazon VPC fournit plusieurs options de connectivité réseau que vous pouvez utiliser, en fonction de vos conceptions et exigences de réseau actuelles. Ces options de connectivité incluent l'utilisation d'Internet ou d'une connexion AWS Direct Connect comme épine dorsale du réseau et l'arrêt de la connexion à AWS ou à des points de terminaison de réseau gérés par l'utilisateur. De plus, avec AWS, vous pouvez choisir la manière dont le routage réseau est fourni entre Amazon VPC et vos réseaux, en tirant parti des services AWS ou de l'équipement et des itinéraires réseau gérés par l'utilisateur. Ce livre blanc examine les options suivantes avec une vue d'ensemble et une comparaison de haut niveau de chacune:

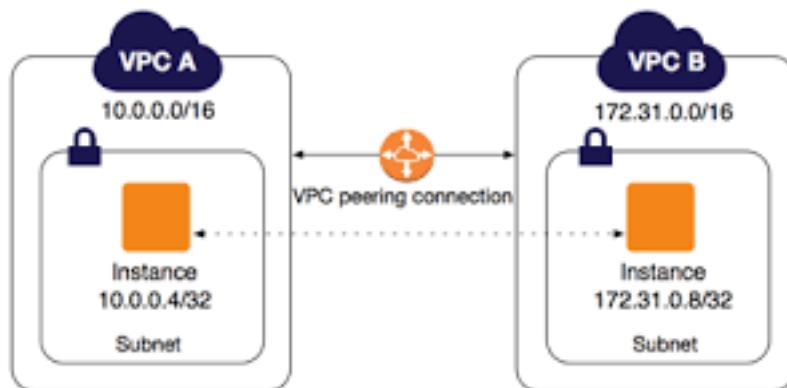


Figure 2.10: Virtual Private Cloud

3.3 Atténuation du déni de service distribué (DDoS):

AWS Shield est un nouveau service géré qui protège vos applications Web contre les attaques DDoS (Distributed Denial of Service). Il fonctionne en conjonction avec Elastic Load Balancing, Amazon CloudFront et Amazon Route 53 et vous protège des attaques DDoS de nombreux types, formes et tailles. Il existe deux niveaux de service:

AWS Shield Standard est disponible pour tous les clients AWS sans frais supplémentaires. Il vous protège de 96 pour cents des attaques les plus courantes aujourd’hui, y compris les inondations SYN / ACK, les attaques par réflexion et les lectures lentes HTTP. Cette protection est appliquée automatiquement et de manière transparente à vos Elastic Load Balancers, vos distributions CloudFront et vos ressources Route 53.



Figure 2.11: AWS Shield

2.3 Développement web sur AWS:

1. Création d'une application React Full-Stack:

1.1 Création d'une application React Full-Stack:

Le moyen le plus simple de créer une application React consiste à utiliser la commande `create-react-app`. Installez ce package à l'aide de la commande suivante dans votre invite de commande ou votre terminal:

Initialisez git et poussez l'application vers le nouveau référentiel GitHub en exécutant les commandes suivantes dans votre interface de ligne de commande:

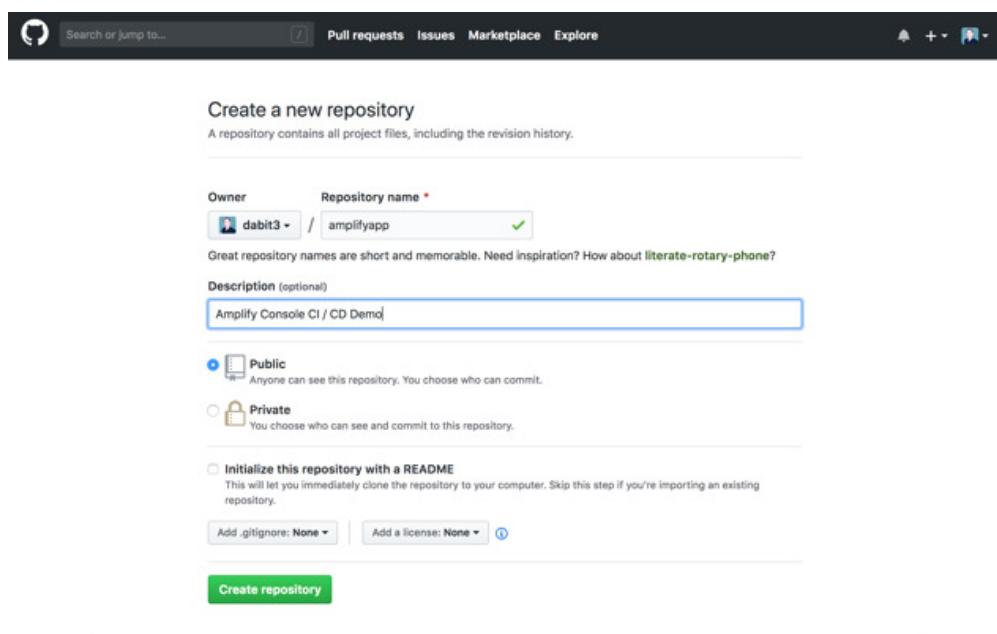


Figure 2.12: Connect to Github Repository

Ouvrez AWS Management Console dans une nouvelle fenêtre de navigateur afin de pouvoir conserver ce guide étape par étape ouvert. Lorsque l'écran se charge, entrez votre nom d'utilisateur et votre mot de passe pour commencer. Tapez ensuite «Amplify» dans la barre de recherche et sélectionnez AWS Amplify pour ouvrir la console de service.

Dans cette étape, vous connecterez le référentiel GitHub que vous venez de créer au service AWS Amplify. Cela vous permettra de créer, déployer et héberger votre application sur AWS.

Dans la console du service AWS Amplify, sélectionnez «Get Started» sous Deploy.

Sélectionnez GitHub comme service de référentiel et sélectionnez Continuer.

Authentifiez-vous avec GitHub et revenez à la console Amplify. Choisissez le référentiel et la branche principale que vous avez créés précédemment, puis sélectionnez Suivant.

Acceptez les paramètres de construction par défaut et sélectionnez Suivant.

Passez en revue les derniers détails et sélectionnez Enregistrer et déployer.

AWS Amplify va maintenant créer votre code source et déployer votre application sur <https://...amplifyapp.com>.

Once the build completes, select the thumbnail to see your web app up and running live.

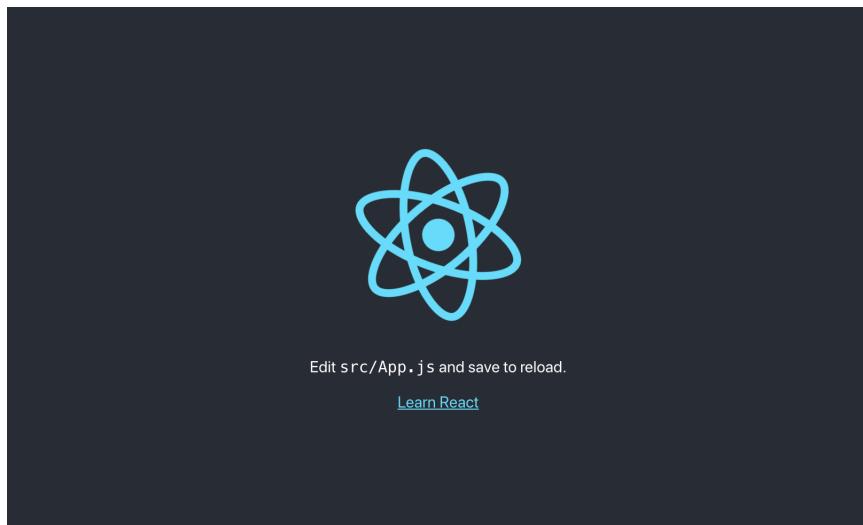


Figure 2.13: React Application

1.2 Création d'initialiser une application Amplify locale:

L’interface de ligne de commande Amplify (CLI) est une chaîne d’outils unifiée permettant de créer des services cloud AWS pour votre application, en suivant un flux de travail guidé simple. Allons-y et installons la CLI Amplify à l’aide de l’invite de commande (Windows) ou du terminal (macOS). REMARQUE: cette commande peut être exécutée dans n’importe quel répertoire de votre invite de commande / terminal car le «-g» indique que le binaire sera installé globalement sur votre système.

npm install -g @aws-amplify/cli

Amazon IAM (Identity and Access Management) vous permet de gérer les utilisateurs et les autorisations utilisateur dans AWS. La CLI utilise IAM pour créer et gérer des services par programmation en votre nom via la CLI.

amplify configure

Dans la console Amplify, cliquez sur Environnements backend.

Dans l'onglet Environnement du backend, copiez la commande amplify init sur votre clavier.

Initialisez le projet Amplify localement avec les commandes ci-dessous.

- ? Enter a name for the project: amplifyapp
- ? Enter a name for the environment: dev
- ? Choose your default editor: Visual Studio Code
- ? Choose the type of app that you're building: javascript
- ? What javascript framework are you using: react
- ? Source Directory Path: src
- ? Distribution Directory Path: build
- ? Build Command: npm run-script build
- ? Start Command: npm run-script start
- ? Do you want to use an AWS profile? Y
- ? Please choose the profile you want to use: your-aws-profile

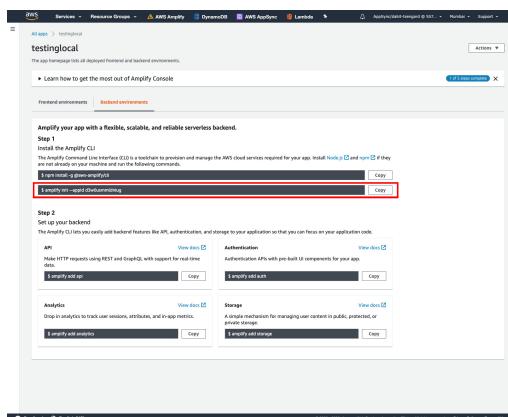


Figure 2.14: AWS Amplify configuration

1.3 AJOUTER UNE AUTHENTIFICATION:

Nous aurons besoin de 2 bibliothèques Amplify pour notre projet. La bibliothèque principale aws-amplify contient toutes les API côté client pour interagir avec les différents services AWS avec lesquels nous allons travailler et la bibliothèque @aws-amplify / ui-react contient des composants d'interface utilisateur spécifiques au framework. Veuillez installer ces bibliothèques à la racine du projet.

npm install aws-amplify @aws-amplify/ui-react

Maintenant que le service d'authentification a été configuré localement, nous pouvons le déployer en exécutant la commande push Amplify:

amplify push -y

La CLI a créé et continuera de mettre à jour un fichier appelé aws-exports.js situé dans le répertoire src de notre projet. Nous utiliserons ce fichier pour informer le projet React des différentes ressources AWS disponibles dans notre projet Amplify.

Pour configurer notre application avec ces ressources, ouvrez src / index.js et ajoutez le code suivant sous la dernière importation:

```
import Amplify from 'aws-amplify'; import config from './aws-exports';
Amplify.configure(config);
```

Ensuite, ouvrez src / App.js et mettez à jour avec le code suivant:

```
import React from 'react'; import logo from './logo.svg'; import './App.css'; import
withAuthenticator, AmplifySignOut from '@aws-amplify/ui-react'

function App()

return (
```

```
<div className="App"><header><img src={logo} className="App-logo" alt="logo"
/><h1>We now have Auth!</h1></header><AmplifySignOut /> </div> ); export
default withAuthenticator(App);
```

Ensuite, exécutez l'application pour voir le nouveau flux d'authentification protégeant l'application: **>npm start >amplify console**

Cela ouvrira la console Amplify dans AWS. Dans la barre latérale de navigation, choisissez Paramètres de l'application > Paramètres de construction et modifiez-la pour ajouter la section backend (lignes 2 à 7 dans le code ci-dessous) à votre amplify.yml. Après avoir effectué les modifications, choisissez Enregistrer.

Ensuite, mettez à jour votre branche frontale pour qu'elle pointe vers l'environnement backend que vous venez de créer. Sous le nom de la branche, choisissez Modifier, puis pointez votre branche principale vers le backend de développement que vous venez de créer. Choisissez Enregistrer.

Revenez maintenant à la fenêtre de votre terminal local et déployez les modifications sur GitHub pour lancer une nouvelle version dans la console Amplify:

```
>git add >git commit -m "added auth" >git push origin master
```

1.4 AJOUTER UNE API ET UNE BASE DE DONNÉES:

Ajoutez une API GraphQL à votre application en exécutant la commande suivante à partir de la racine de votre répertoire d'application:

- ? Please select from one of the below mentioned services: GraphQL
- ? Provide API name: notesapp
- ? Choose the default authorization type for the API: API Key
- ? Enter a description for the API key: demo
- ? After how many days from now the API key should expire: 7 (or your preferred expiration)
- ? Do you want to configure advanced settings for the GraphQL API: No, I am done.
- ? Do you have an annotated GraphQL schema? No
- ? Do you want a guided schema creation? Yes
- ? What best describes your project: Single object with fields
- ? Do you want to edit the schema now? Yes

Ouvrez le schéma GraphQL dans votre éditeur de texte: amplify / backend / api / myapi / schema.graphql. Mettez à jour le fichier avec le schéma suivant:

```
>type Note @model id: ID! name: String! description: String
```

Enregistrez le fichier. Revenez ensuite à la ligne de commande et appuyez sur Entrée pour terminer l'étape de configuration de l'API.

Maintenant que l'API a été configurée localement, il est temps de la déployer. Pour ce faire, exécutez la commande push Amplify:

```
amplify push -y
```

Cela fera 3 choses:

Créer l'API AppSync
 Créer une table DynamoDB
 Créez les opérations GraphQL locales dans un dossier situé dans src / graphql que vous pouvez utiliser pour interroger l'API
 To view the GraphQL API in your account at any time, run the following command:

> Choose GraphQL

Pour afficher l'application Amplify dans votre compte à tout moment, exécutez la commande suivante: amplify console

Maintenant que le back-end a été déployé, écrivons du code pour permettre aux utilisateurs de créer, répertorier et supprimer des notes.

Mettez à jour src / App.js avec le code suivant: There are 3 main functions in our app: fetchNotes - This function uses the API class to send a query to the GraphQL API and retrieve a list of notes. createNote - This function also uses the API class to send a mutation to the GraphQL API, the main difference is that in this function we are passing in the variables needed for a GraphQL mutation so that we can create a new note with the form data. deleteNote - Like createNote, this function is sending a GraphQL mutation along with some variables, but instead of creating a note we are deleting a note.

Pour tester l'application, exécutez la commande de démarrage: **npm start**

1.5 AJOUTER UN STOCKAGE:

Pour ajouter une fonctionnalité de stockage d'images, nous utiliserons la catégorie de stockage Amplify:

- ? Please select from one of the below mentioned services: Content
- ? Please provide a friendly name for your resource that will be used to label this category in the project: imagestorage
- ? Please provide bucket name: <your-unique-bucket-name>
- ? Who should have access: Auth users only
- ? What kind of access do you want for Authenticated users? create, read, update, delete
- ? Do you want to add a Lambda Trigger for your S3 Bucket? N

Next, open amplify/backend/api/notesapp/schema.graphql and update it with the following schema:

```
type Note @model id: ID! name: String! description: String image: String
```

Maintenant que le service de stockage a été configuré localement et que nous avons mis à jour le schéma GraphQL, nous pouvons déployer les mises à jour en exécutant la commande push Amplify:

amplify push -y

Maintenant que le backend a été mis à jour, mettons à jour l'application React pour ajouter la fonctionnalité permettant de télécharger et d'afficher des images pour une note. Ouvrez src / App.js et apportez les modifications suivantes.

- Ajoutez d'abord la classe Storage à vos importations Amplify:

```
import API, Storage from 'aws-amplify';
```

- Dans la fonction principale de l'application, créez une nouvelle fonction onChange pour gérer le téléchargement de l'image:

```
async function onChange(e)
if (!e.target.files[0]) return
const file = e.target.files[0];
setFormData( ...formData, image: file.name );
await Storage.put(file.name, file);
fetchNotes();
```

- Mettez à jour la fonction fetchNotes pour récupérer une image si une image est associée à une note:

```
async function fetchNotes()
const apiData = await API.graphql( query: listNotes );
const notesFromAPI = apiData.data.listNotes.items;
await Promise.all(notesFromAPI.map(async note =>
if (note.image)
const image = await Storage.get(note.image);
note.image = image;
return note;))
```

```
setNotes(apiData.data.listNotes.items);
```

d. Mettez à jour la fonction `createNote` pour ajouter l'image au tableau d'images local si une image est associée à la note:

```
async function createNote()
{
    if (!formData.name || !formData.description) return;

    await API.graphql( query: createNoteMutation, variables: { input: formData } );

    if (formData.image)
        const image = await Storage.get(formData.image);

    formData.image = image;
    setNotes([ ...notes, formData ]);
    setFormData(initialFormState);
}
```

e. Ajoutez une entrée supplémentaire au formulaire dans le bloc de retour:

```
<input type="file" onChange={onChange} />
```

F. Lors du mappage sur le tableau de notes, rendez une image si elle existe:

Pour tester l'application, exécutez la commande de démarrage:

```
npm start
```

2.4 Machine Learning sur AWS:

1. Train a Deep Learning model with AWS Deep Learning Containers on Amazon EC2:

Les conteneurs AWS Deep Learning (DL Containers) sont des images Docker préinstallées avec des frameworks d'apprentissage en profondeur pour faciliter le déploiement rapide d'environnements d'apprentissage automatique personnalisés en vous permettant d'éviter le processus complexe de création et d'optimisation de vos environnements à partir de zéro.

À l'aide des conteneurs AWS DL, les développeurs et les spécialistes des données peuvent rapidement ajouter l'apprentissage automatique à leurs applications conteneurisées déployées sur Amazon Elastic Container Service pour Kubernetes (Amazon EKS), Kubernetes autogéré, Amazon Elastic Container Service (Amazon ECS) et Amazon EC2.

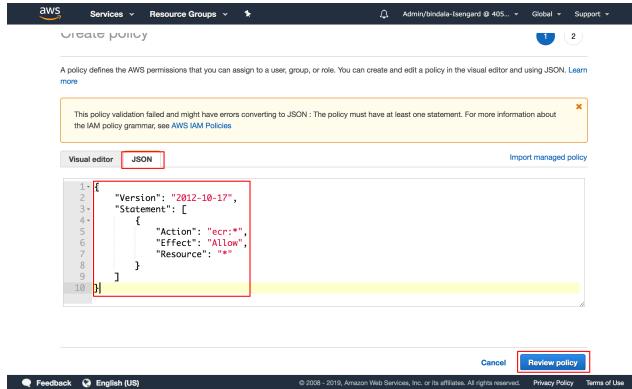


Figure 2.15: AWS ECR Permission

1. Sign-up for AWS
2. Add permissions for accessing Amazon ECR

Les images AWS Deep Learning Container sont hébergées sur Amazon Elastic Container Registry (ECR), un registre de conteneurs Docker entièrement géré qui permet aux développeurs de stocker, gérer et déployer facilement des images de conteneurs Docker. Dans cette étape, vous accorderez à un utilisateur IAM existant des autorisations pour accéder à Amazon ECR (à l'aide de la stratégie `AmazonECSFullAccess`).

3. Launch an AWS Deep Learning Base AMI instance: Choisissez l'onglet AWS Marketplace sur la gauche, puis recherchez «deep learning base ubuntu». Sélectionnez Deep Learning Base AMI (Ubuntu). Vous pouvez également sélectionner l'AMI Deep Learning Base (Amazon Linux).

utilisez une instance c5.large, mais vous pouvez choisir des types d'instances supplémentaires, y compris les instances P3 basées sur GPU.

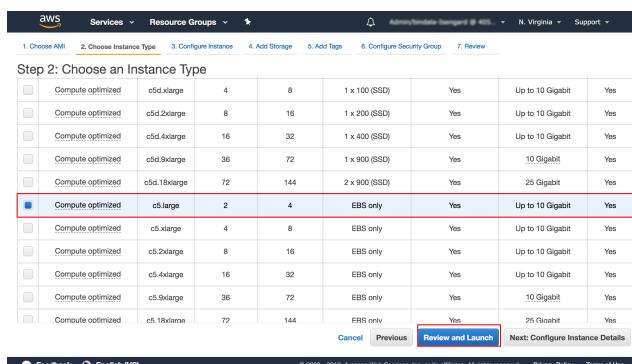


Figure 2.16: ML Instance type

4. Connect to your instance

```
cd /Users/<your_username> /Downloads/
chmod 0400 <your .pem filename>
ssh -L localhost:8888:localhost:8888 -i <your .pem filename> ubuntu@<your instance
DNS>
```

```
1. ubuntu@ip-172-31-6-225: ~ (ssh)
:~ bindala$ cd /Users/      /Downloads/
:Downloads      $ chmod 0400 my_new_key_pair_name.pem
:Downloads      $ ssh -L localhost:8888:localhost:8888 -i my_new_key_pair_name.pem ubuntu@ec2-35-196-16.compute-1.amazonaws.com
The authenticity of host 'ec2-35-196-16.compute-1.amazonaws.com (35.175.196.16)' can't be established.
ECDSA key fingerprint is SHA256:yeAcMUx2oosCKcQm+VnKAQ9bdswo2lSYu87r6ph5BSs.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-35-196-16.compute-1.amazonaws.com,35.175.196.16' (ECDSA) to the list of known hosts.
bind: Address already in use
channel_setup_fwd_listener_tcpip: cannot listen to port: 8888
Could not request local forwarding.
=====
 _I _I_ )
 _I (   / Deep Learning Base AMI (Ubuntu) Version
 ___\_\_I___| =====
=====
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1077-aws x86_64v)

Nvidia driver version: 410.104
CUDA versions available: cuda-10.0 cuda-8.0 cuda-9.0 cuda-9.2
Default CUDA version is 9.0
Libraries: cuDNN, NCCL, Intel MKL-DNN

AWS Deep Learning AMI Homepage: https://aws.amazon.com/machine-learning/amis/
Developer Guide and Release Notes: https://docs.aws.amazon.com/dlami/latest/devguide/what-is-dlami.html
Support: https://forums.aws.amazon.com/forum.jspa?forumID=263
```

Figure 2.17: Connect to the instance

5. Log in to Amazon ECR

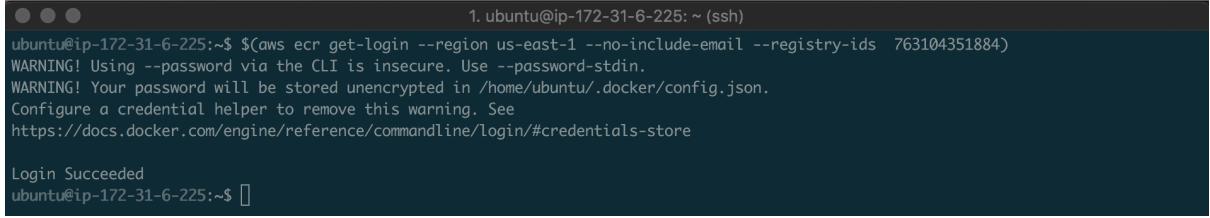
Les images AWS Deep Learning Container sont hébergées sur Amazon Elastic Container Registry (ECR), un registre de conteneurs Docker entièrement géré qui permet aux développeurs de stocker, gérer et déployer facilement des images de conteneurs Docker. Dans cette étape, vous vous connecterez et vérifierez l'accès à Amazon ECR.

Vous devez fournir votre ID de clé d'accès AWS et votre clé d'accès secrète. Si vous ne disposez pas déjà de ces informations, vous pouvez créer un ID de clé d'accès et une clé d'accès secrète ici.

```
1. ubuntu@ip-172-31-6-225: ~ (ssh)
ubuntu@ip-172-31-6-225:~$ aws configure
AWS Access Key ID [None]: [REDACTED]
AWS Secret Access Key [None]: [REDACTED]
Default region name [None]: [REDACTED]
Default output format [None]: [REDACTED]
```

Figure 2.18: AWS ECR Access Verification

Log in to Amazon ECR:



```
ubuntu@ip-172-31-6-225:~$ $(aws ecr get-login --region us-east-1 --no-include-email --registry-ids 763104351884)
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ubuntu@ip-172-31-6-225:~$ 
```

Figure 2.19: Login to AWS ECR

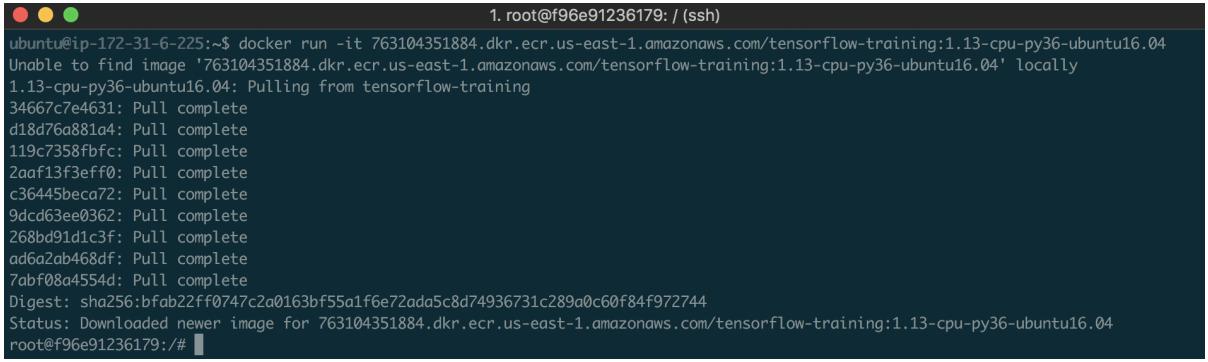
(aws ecr get-login –region us-east-1 –no-include-email –registry-ids 763104351884)

6. Run TensorFlow training with Deep Learning Containers

Nous utiliserons une image AWS Deep Learning Container pour la formation TensorFlow sur des instances de CPU avec Python 3.6.

a. Run AWS Deep Learning Containers

docker run -it 763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training:1.13-cpu-py36-ubuntu16.04



```
root@f96e91236179:~$ docker run -it 763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training:1.13-cpu-py36-ubuntu16.04
Unable to find image '763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training:1.13-cpu-py36-ubuntu16.04' locally
1.13-cpu-py36-ubuntu16.04: Pulling from tensorflow-training
34667c7e4631: Pull complete
d18d76a881a4: Pull complete
119c7358fbfc: Pull complete
2aa13f3eff0: Pull complete
c36445beca72: Pull complete
9dcfd3ee0362: Pull complete
268bd91d1c3f: Pull complete
ad6a2ab468df: Pull complete
7abf08a4554d: Pull complete
Digest: sha256:bfbab22ff0747c2a0163bf55a1f6e72ada5c8d74936731c289a0c60f84f972744
Status: Downloaded newer image for 763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training:1.13-cpu-py36-ubuntu16.04
root@f96e91236179:~# 
```

Figure 2.20: Run AWS DL Containers

b. Pull an example model to train

git clone https://github.com/fchollet/keras.git

c. Start training

python keras/examples/mnist_cnn.py

7. Terminate Your Resources: vous mettrez fin à l'instance Amazon EC2 que vous avez créée.

```

● ● ●
root@f96e91236179:/# python keras/examples/mnist_cnn.py
Using TensorFlow backend.
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz
11493376/11490434 [=====] - 0s 0us/step
x_train shape: (60000, 28, 28, 1)
60000 train samples
10000 test samples
WARNING:tensorflow:From /usr/local/lib/python3.6/site-packages/tensorflow/python/framework/op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use 'rate' instead of 'keep_prob'. Rate should be set to 'rate = 1 - keep_prob'.
WARNING:tensorflow:From /usr/local/lib/python3.6/site-packages/tensorflow/python/ops/math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 60000 samples, validate on 10000 samples
Epoch 1/12
2019-03-26 05:32:01.094551: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX512F
2019-03-26 05:32:01.099006: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 3000000000 Hz
2019-03-26 05:32:01.099226: I tensorflow/compiler/xla/service/service.cc:150] XLA service 0x37adb70 executing computations on platform Host. Devices:
2019-03-26 05:32:01.099253: I tensorflow/compiler/xla/service/service.cc:158] StreamExecutor device (0): <undefined>, <undefined>
2019-03-26 05:32:01.100002: I tensorflow/core/common_runtime/process_util.cc:71] Creating new thread pool with default inter op setting
: 2. Tune using inter_op_parallelism_THREADS for best performance.
60000/60000 [=====] - 72s 1ms/step - loss: 0.2733 - acc: 0.9154 - val_loss: 0.0616 - val_acc: 0.9801
Epoch 2/12
60000/60000 [=====] - 70s 1ms/step - loss: 0.0883 - acc: 0.9736 - val_loss: 0.0467 - val_acc: 0.9844
Epoch 3/12
5760/60000 [=> .....] - ETA: 1:01 - loss: 0.0730 - acc: 0.9792

```

Figure 2.21: Running the DL model

2. Amazon SageMaker:

Amazon SageMaker est un service entièrement géré permettant aux développeurs et aux scientifiques des données de créer, de former et de déployer rapidement et facilement des modèles de machine learning (ML). SageMaker facilite chaque étape du processus de machine learning afin de rendre plus aisés le développement de modèles de haute qualité.

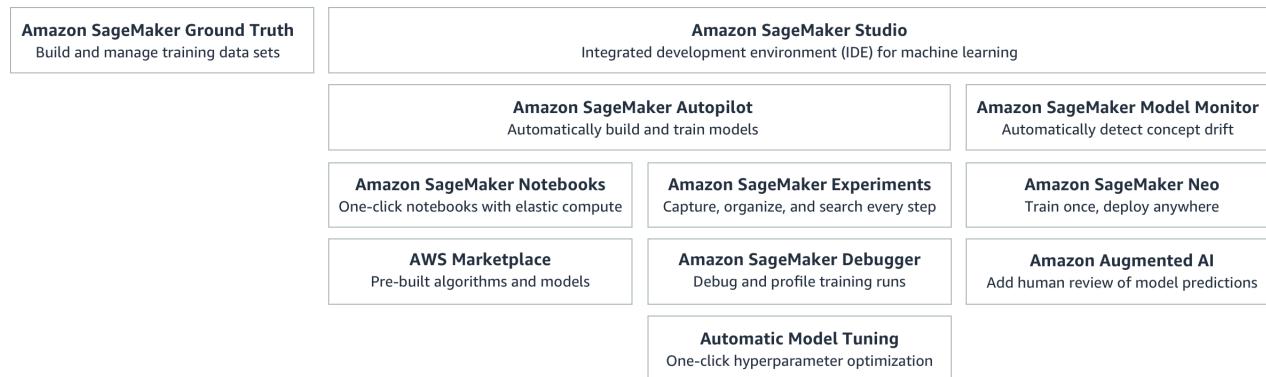


Figure 2.22: SageMaker Services Architecture

Toutes les activités de développement ML, y compris les blocs-notes, la gestion des expériences, la création automatique de modèles, le débogage et la détection de dérive de modèles peuvent être effectuées dans l'interface visuelle unifiée de SageMaker Studio.

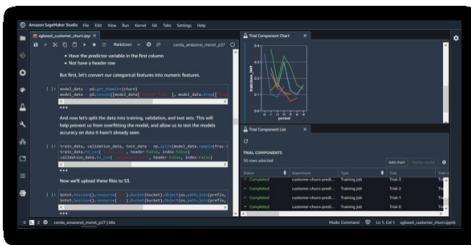


Figure 2.23: SageMaker Studio

Services IA sur AWS:

Les services IA pré-entraînés fournissent des informations prêtées à l'emploi pour vos applications et vos flux de travail afin de vous aider à améliorer les résultats de votre entreprise. Ils sont basés sur la même technologie que celle utilisée par Amazon pour ses propres activités. Vous pouvez créer des applications optimisées par l'IA sans aucune expertise en matière de machine learning.

1. Amazon Comprehend est un service de traitement du langage naturel qui exploite l'apprentissage automatique pour identifier des informations et des relations dans un texte. Aucune expérience de l'apprentissage automatique n'est requise.
2. Amazon CodeGuru est un outil pour développeurs optimisé par le machine learning qui fournit des recommandations intelligentes afin d'améliorer la qualité du code et d'identifier les lignes de code les plus onéreuses d'une application.
3. Amazon Lex est un service permettant de créer des interfaces de conversation dans une application reposant sur la voix et le texte. Amazon Lex offre des fonctionnalités d'apprentissage profond avancées.
4. Amazon Forecast est un service entièrement géré qui utilise le machine learning pour fournir des prévisions extrêmement précises.
5. Amazon Textract est un service de machine learning entièrement géré qui extrait automatiquement du texte et des données à partir de documents numérisés.
6. Amazon Fraud Detector est un service entièrement géré qui utilise le machine learning (ML) qui bénéficie de plus de 20 ans d'expertise d'Amazon en matière de détection des fraudes, pour identifier les activités potentiellement frauduleuses afin que les clients puissent détecter plus rapidement les fraudes en ligne.
7. Amazon Rekognition facilite l'ajout d'analyses d'images et de vidéos à vos applications à l'aide d'une technologie de deep learning éprouvée, hautement évolutive et qui ne nécessite aucune expertise en machine learning. Avec Amazon Rekognition, vous pouvez identifier des objets, des personnes, du texte, des scènes et des activités dans des images et des vidéos, ainsi que détecter le contenu inapproprié. Amazon Rekognition fournit également des fonctionnalités d'analyse du visage et de recherche faciale extrêmement précises que vous pouvez utiliser pour détecter, analyser et comparer les visages dans un large éventail de cas d'utilisation : vérification des utilisateurs, comptage des personnes, sécurité publique, etc.

Chapter 3

Introduction à DevOps sur AWS

DevOps est un nouveau terme qui se concentre principalement sur l'amélioration de la collaboration, de la communication, et l'intégration entre les développeurs de logiciels et les opérations informatiques. C'est un terme générique que certains décrivent comme une philosophie, un changement culturel et un changement de paradigme.

Le déploiement de logiciels a été principalement le rôle du groupe des opérations informatiques. Fondamentalement, les développeurs aiment créer des logiciels et changer les choses rapidement, alors que l'informatique les opérations se concentrent sur la stabilité et la fiabilité. Cette inadéquation des objectifs peut conduire à des conflits, et finalement l'entreprise peut en souffrir.

Aujourd'hui, ces anciennes divisions s'effondrent, avec la fusion des rôles IT et développeur et suivant une série de principes systématiques:

- Infrastructure en tant que code
- Déploiement continu
- Automatisation
- Surveillance
- Sécurité

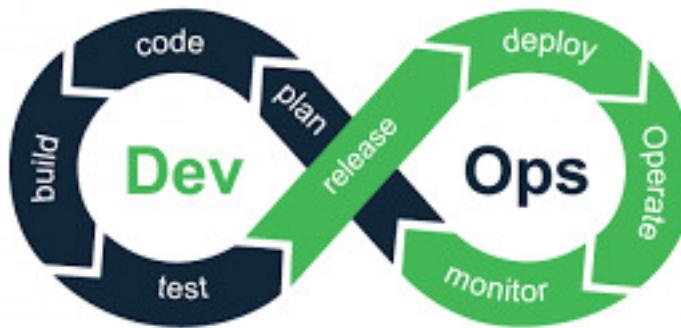


Figure 3.1: DevOps Démarche

3.1 Agile Evolution to DevOps

L'histoire commence par le développement logiciel agile, qui est devenu populaire il y a plus de dix ans et était considérée comme une meilleure approche de la création de logiciels. Avant d'être agile, la méthodologie dominante de développement de la cascade était basée sur une séquence commençant avec une phase d'exigences où 100

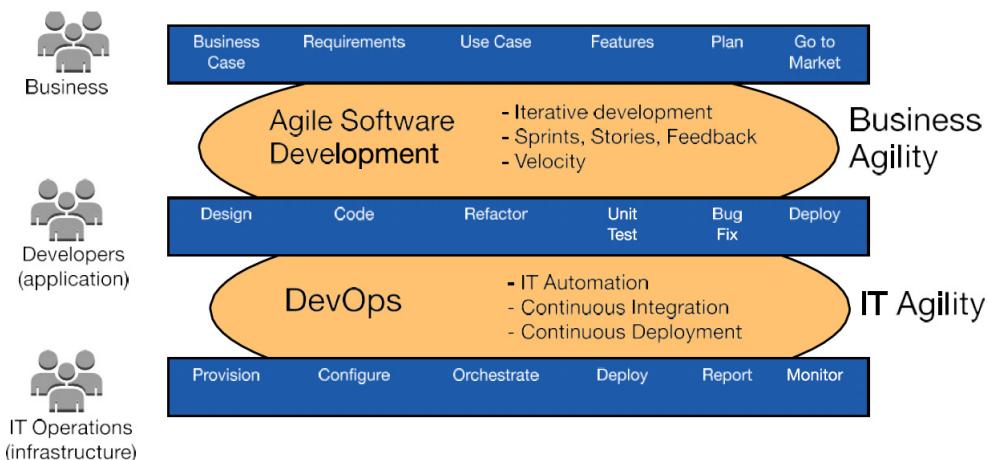


Figure 3.2: DevOps-Agile

Au cours des dernières années, l'évolution du développement logiciel agile a commencé à bouger en aval vers l'infrastructure sous le label DevOps. Alors que les logiciels agiles le développement se concentre principalement sur la collaboration entre l'entreprise et ses développeurs, DevOps se concentre sur la collaboration entre développeurs, opérations informatiques et les équipes de sécurité. Les opérations informatiques incluent les administrateurs système, la base de données administrateurs, ingénieurs réseau, architectes d'infrastructure et personnel de support. Alors que le développement logiciel agile offre l'agilité commerciale, DevOps fournit agilité, permettant le déploiement d'applications plus fiables, prévisibles et efficaces

Les pratiques DevOps varient en fonction de la tâche: avec le développement d'applications, DevOps se concentre sur création de code, couverture de code, tests unitaires, empaquetage et déploiement. Avec infrastructure, d'autre part, DevOps se concentre sur l'approvisionnement, la configuration, orchestration et déploiement. Mais dans chaque domaine, les principes sous-jacents de la version la gestion, le déploiement, la restauration, la restauration avant et les tests restent les mêmes.

3.2 Infrastructure en tant que code

Pratiquer «l'infrastructure en tant que code» signifie appliquer la même rigueur de code d'application du développement à l'approvisionnement des infrastructures. Toutes les configurations doivent être définies de manière déclarative et stockées dans un système de gestion de version, tout comme le code d'application. Le provisionnement, l'orchestration et le déploiement de l'infrastructure doivent prendre en charge l'utilisation du «Code d'infrastructure».

Jusqu'à récemment, la rigueur appliquée au développement du code d'application n'a pas nécessairement été appliquée aux infrastructures. L'infrastructure est fréquemment provisionnée à l'aide du manuel processus. Les scripts développés pendant l'approvisionnement peuvent ne pas être stockés dans la version les systèmes de contrôle et la création d'environnements ne sont pas toujours reproductibles, fiables ou cohérents.

En revanche, AWS fournit un moyen axé sur le DevOps de créer et de maintenir Infrastructure. Similaire à la façon dont les développeurs de logiciels écrivent le code d'application, AWS fournit des services qui permettent la création, le déploiement et la maintenance de infrastructure de manière programmatique, descriptive et déclarative. Ces services offrent rigueur, clarté et fiabilité. Les services AWS abordés dans ce document sont essentiels à une stratégie DevOps et constituent les fondements de nombreux AWS de niveau supérieur Principes et pratiques DevOps.

Un bon exemple de la façon dont les principes DevOps sont utilisés dans la pratique est AWS CloudFormation. En utilisant les modèles AWS CloudFormation, vous pouvez définir et modéliser Ressources AWS qui peuvent être créées et mises à jour. Vous pouvez créer des modèles AWS CloudFormation personnalisés ou utiliser des exemples de modèles sont accessibles au public.

Avec les modèles, vous pouvez travailler avec un large éventail d'offres AWS, y compris Amazon Service de stockage simple (Amazon S3), Auto Scaling, Amazon CloudFront, Amazon DynamoDB, Amazon Elastic Compute Cloud (EC2), Amazon ElastiCache, AWS Elastic Beanstalk, Elastic Load Balancing, AWS Identity and Access Management, AWS OpsWorks et Amazon Virtual Private Cloud.

AWS CloudFormation facilite l'organisation et le déploiement d'une collection d'AWS ressources et vous permet de décrire les dépendances ou de transmettre des paramètres spéciaux lorsque la pile est configurée.

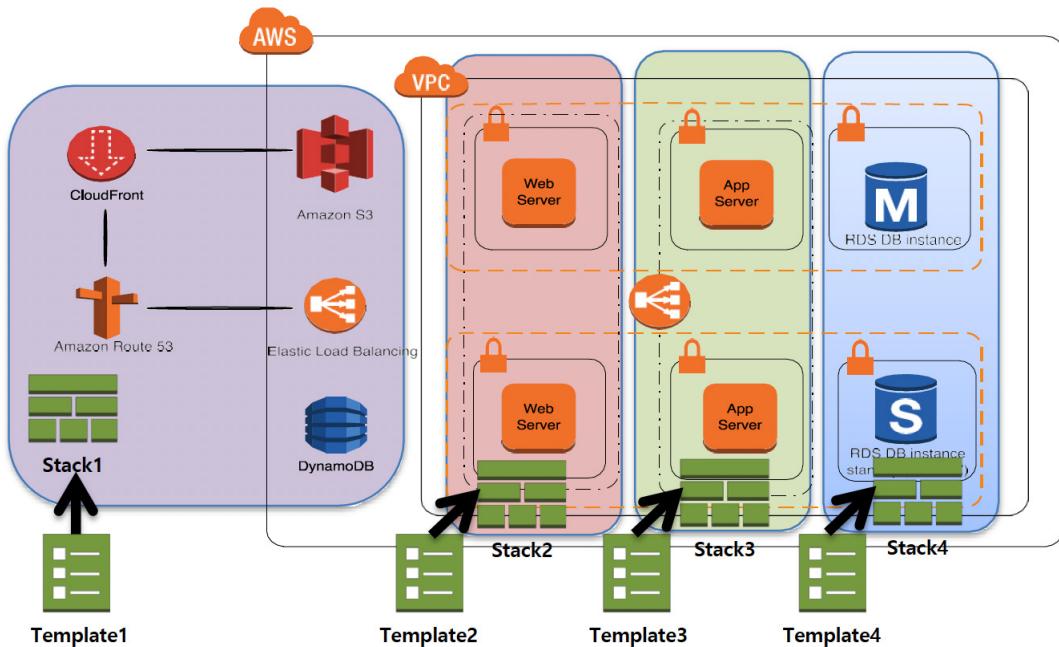


Figure 3.3: AWS CloudFormation example

```
{
  "Description": "Create an EC2 instance running the Amazon Linux 32 bit AMI.",
  "Parameters": {
    "KeyPair": {
      "Description": "The EC2 Key Pair to allow SSH access to the instance",
      "Type": "String"
    }
  },
  "Resources": {
    "Ec2Instance": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
        "KeyName": { "Ref": "KeyPair" },
        "ImageId": "ami-3b355a52"
      }
    }
  },
  "Outputs": {
    "InstanceId": {
      "Description": "The InstanceId of the newly created EC2 instance",
      "Value": { "Ref": "Ec2Instance" }
    }
  },
  "AWSTemplateFormatVersion": "2010-09-09"
}
```

Figure 3.4: AWS CloudFormation template for launching an EC2 instance

Pour réaliser le potentiel d’AWS CloudFormation en matière d ”informations sous forme de code”, vous devez stocker modèles dans un contrôle de version des systèmes de gestion de code source avant de déployer ou mettez-les à jour dans AWS. Amazon S3 fournit un bon emplacement pour le stockage et la gestion des versions modèles. Vous pouvez intégrer AWS CloudFormation au développement et outils de gestion de votre choix.

Amazon Machine Image (AMI):

Une Amazon Machine Image (AMI) est un autre exemple d ’«infrastructure en tant que code». Ce Le composant principal d’AWS computing est une sorte de modèle numérique qui peut lancer (mise à disposition) d’instances Amazon EC2, l’environnement de calcul AWS fondamental dans le nuage. L’image contient une configuration logicielle telle qu’un serveur Web, une application serveur et base de données.

3.3 Déploiement continu

Le déploiement continu est un autre concept clé dans une stratégie DevOps. Son objectif principal est pour permettre le déploiement automatisé du code d’application prêt pour la production.

En utilisant des pratiques et des outils de livraison continue, les logiciels peuvent être déployés rapidement, à plusieurs reprises et de manière fiable. Si un déploiement échoue, il peut être automatiquement restauré la version précédente.

AWS CodeDeploy:

Un excellent exemple de ce principe dans AWS est le service de déploiement de code AWS CodeDeploy.

Voici comment ça fonctionne:

1. Le contenu de l’application est emballé et déployé sur Amazon S3 avec un Fichier spécifique à l’application (AppSpec) qui définit une série d’étapes de déploiement qui AWS CodeDeploy doit s’exécuter. Le package est appelé une «révision» CodeDeploy.
2. Vous créez une application dans AWS CodeDeploy et définissez les instances auxquelles le l’application doit être déployée (DeploymentGroup). L’application définit également le Le compartiment Amazon S3 où réside le package de déploiement.
3. Un agent AWS CodeDeploy est déployé sur chaque Amazon EC2 participant. le l’agent interroge AWS CodeDeploy pour déterminer quoi et quand extraire une révision du compartiment Amazon S3 spécifié.
4. L’agent AWS CodeDeploy extrait le code d’application empaqueté et le déploie sur l’instance. Le fichier AppSpec contenant les instructions de déploiement est également téléchargé.

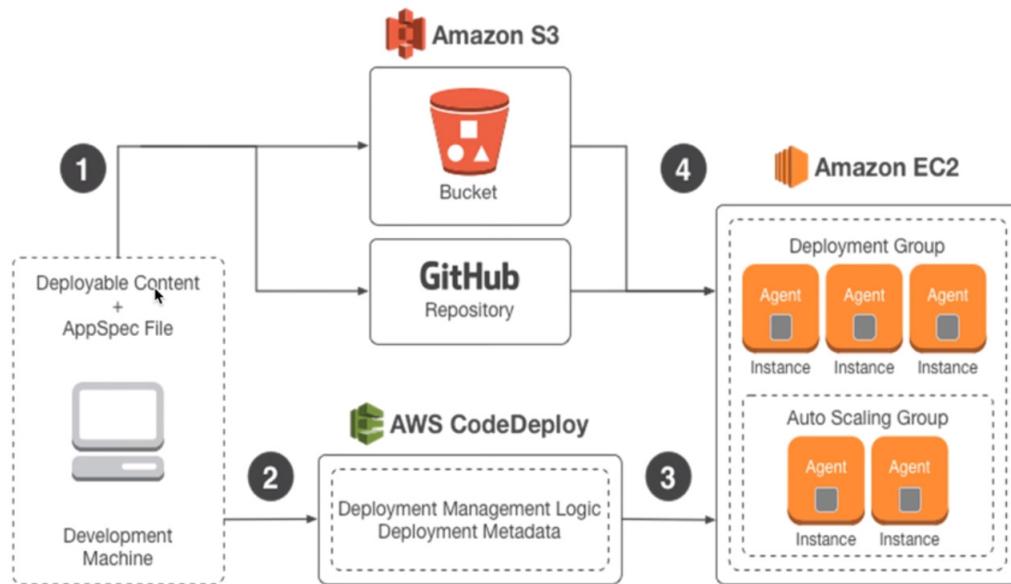


Figure 3.5: AWS CodeDeploy process

De cette manière, AWS CodeDeploy illustre le déploiement automatisé continu qui est au cœur de DevOps.

Comme AWS CodeDeploy, AWS CodePipeline, AWS CodeCommit, AWS Elastic Beanstalk et AWS OpsWorks sont également très efficaces dans le déploiement continu.

3.4 Automatisation:

L'automatisation est une autre philosophie et pratique de base de DevOps.

L'automatisation se concentre sur l'installation, la configuration, le déploiement et le support de l'infrastructure et des applications qui courent dessus. En utilisant l'automatisation, vous pouvez configurer des environnements plus rapidement de manière standardisée et reproductible.

La suppression des processus manuels est la clé d'une stratégie DevOps réussie. L'automatisation présente de nombreux avantages:

- Changements rapides
- Amélioration de la productivité
- Configurations répétables
- Environnements reproductibles
- Élasticité optimisée
- Mise à l'échelle automatique optimisée

- Tests automatisés

L'automatisation est une pierre angulaire des services AWS et est prise en charge en interne dans tous services, fonctionnalités et offres.

AWS OpsWorks:

AWS OpsWorks pousse les principes de DevOps encore plus loin qu'AWS Elastic Beanstalk. Il peut être considéré comme un service de gestion d'applications plutôt que comme un simple conteneur d'application. AWS OpsWorks offre encore plus de niveaux d'automatisation avec des fonctionnalités supplémentaires telles que l'intégration avec le logiciel de gestion de la configuration (Chef) et gestion du cycle de vie des applications. Vous pouvez utiliser la gestion du cycle de vie des applications pour définir quand les ressources sont installées, configurées, déployées, non déployées ou arrêtées.

Les piles d'applications sont organisées en couches architecturales afin que les piles puissent être entretenues indépendamment. Les exemples de couches peuvent inclure le niveau Web, le niveau application et niveau base de données. Prêt à l'emploi, AWS OpsWorks simplifie également la configuration d'Auto Scaling groupes et équilibriseurs de charge Elastic Load Balancing, illustrant davantage le DevOps principe d'automatisation. Tout comme AWS Elastic Beanstalk, AWS OpsWorks prend en charge gestion des versions d'applications, déploiement continu et configuration de l'infrastructure la gestion.

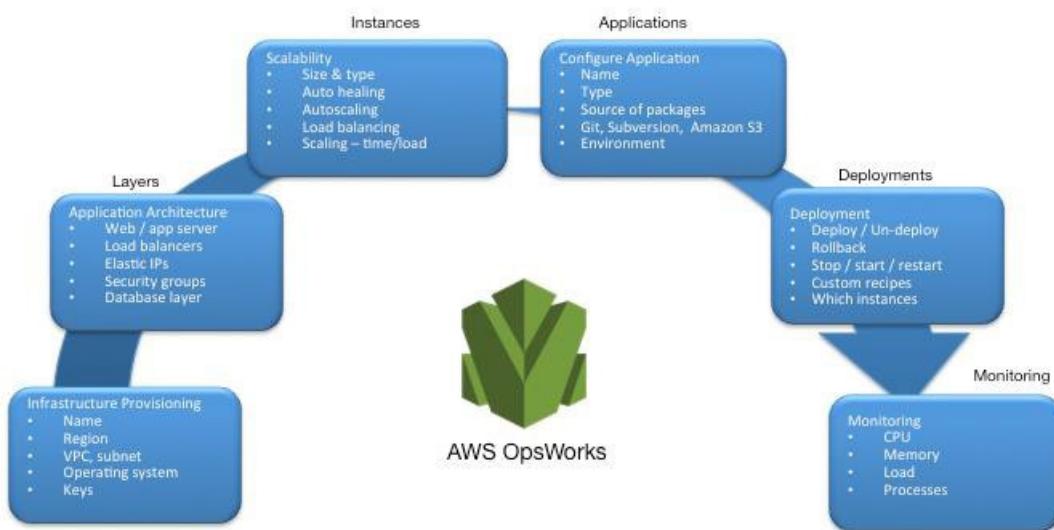


Figure 3.6: AWS OpsWorks showing DevOps features and architecture

3.5 Surveillance

La communication et la collaboration sont fondamentales dans une stratégie DevOps. Pour faciliter cela, la rétroaction est essentielle. Dans AWS, les commentaires sont

fournis par deux services principaux: Amazon CloudWatch et AWS CloudTrail. Ensemble, ils fournissent une surveillance robuste, des alertes, et l'audit de l'infrastructure afin que les développeurs et les équipes opérationnelles puissent travailler ensemble étroitement et de manière transparente.

Amazon CloudWatch:

Amazon CloudWatch surveille en temps réel toutes les ressources AWS et les applications que vous exécutez dessus. Les ressources et les applications peuvent produire des métriques qu'Amazon CloudWatch rassemble et suit. Vous pouvez configurer des alarmes pour envoyer des notifications lorsque des événements se produisent. Vous pouvez configurer les notifications dans de nombreux formats, y compris les e-mails, Amazon SNS et Amazon Simple Queue Service. Les notifications peuvent être envoyées à individus, équipes ou autres ressources AWS.

En plus de fournir des commentaires, Amazon CloudWatch prend également en charge le concept DevOps d'automatisation. Les services AWS tels que Auto Scaling s'appuient sur CloudWatch pour les notifications qui déclenchent une action automatisée appropriée telle que la mise à l'échelle et la réduction d'Amazon Les instances EC2 et la charge augmentent ou diminuent.



Figure 3.7: Example DevOps automation using Amazon CloudWatch and Auto Scaling

3.6 Sécurité

Dans un environnement compatible DevOps, se concentrer sur la sécurité est toujours d'une importance primordiale. L'infrastructure et les actifs de l'entreprise doivent être protégés, et lorsque des problèmes surviennent, ils doivent être traités rapidement et efficacement.

Identity and Access Management (IAM):

Le service AWS Identity and Access Management (IAM) est l'un des composants d'AWS infrastructure de sécurité. Avec IAM, vous pouvez gérer de manière centralisée les utilisateurs et la sécurité les informations d'identification telles que les mots de passe, les clés d'accès et les politiques d'autorisations qui contrôlent Les services et ressources AWS peuvent accéder aux utilisateurs.

Conclusion

AWS fournit des blocs de construction que vous pouvez assembler rapidement pour prendre en charge virtuellement toute charge de travail. Avec AWS, vous trouverez un ensemble complet de services hautement disponibles qui sont conçus pour fonctionner ensemble pour créer des applications évolutives sophistiquées.

Vous avez accès à un stockage hautement durable, à des calculs à faible coût, à des bases de données hautes performances, à des outils de gestion, etc. Tout cela est disponible sans coût initial et vous ne payez que ce que vous utilisez. Ces services aident les organisations évoluer plus rapidement, réduisent les coûts informatiques et évoluent. AWS bénéficie de la confiance des plus grandes entreprises et des start-ups les plus en vogue pour alimenter une grande variété de charges de travail, y compris les applications Web et mobiles, le développement de jeux, les données traitement et entreposage, stockage, archivage et bien d'autres.

Bibliographie

1.AWS Cloud Practitioner Essentials (Second Edition):

https://www.aws.training/Details/Curriculum?id=27076fbclid=IwAR2tuDh-Y2KyBZKL_m4VhBgkp5ZY8gBQmtuNNsxEwcfFG7xHRWvGvzmYKzY

2.Overview of AWS:<https://docs.aws.amazon.com/whitepapers/latest/aws-overview/introduction.html>

3.A Gentle Introduction to Amazon Web Services (AWS):

<https://medium.com/@praneeth.jm/a-gentle-introduction-to-amazon-web-services-aws-50f18c7c57dc>

4.Introduction to DevOps on AWS:

[https://d0.awsstatic.com/whitepapers/AWS_{DevOps}.pdf](https://d0.awsstatic.com/whitepapers/AWS_DevOps.pdf)

5.AWS Cognito:

<https://tutorialslink.com/Articles/What-is-AWS-Cognito-Amazon-Cognito/1616>

6.les didacticiels de prise

en main adaptés à vos besoins AWS: https://aws.amazon.com/fr/getting-started/hands-on/?fbclid=IwAR1bLet7U3NovufbcTWSEVYSCFKK_xI2a9f9h6EKhGtCNHwaHRCjs6q4AxY