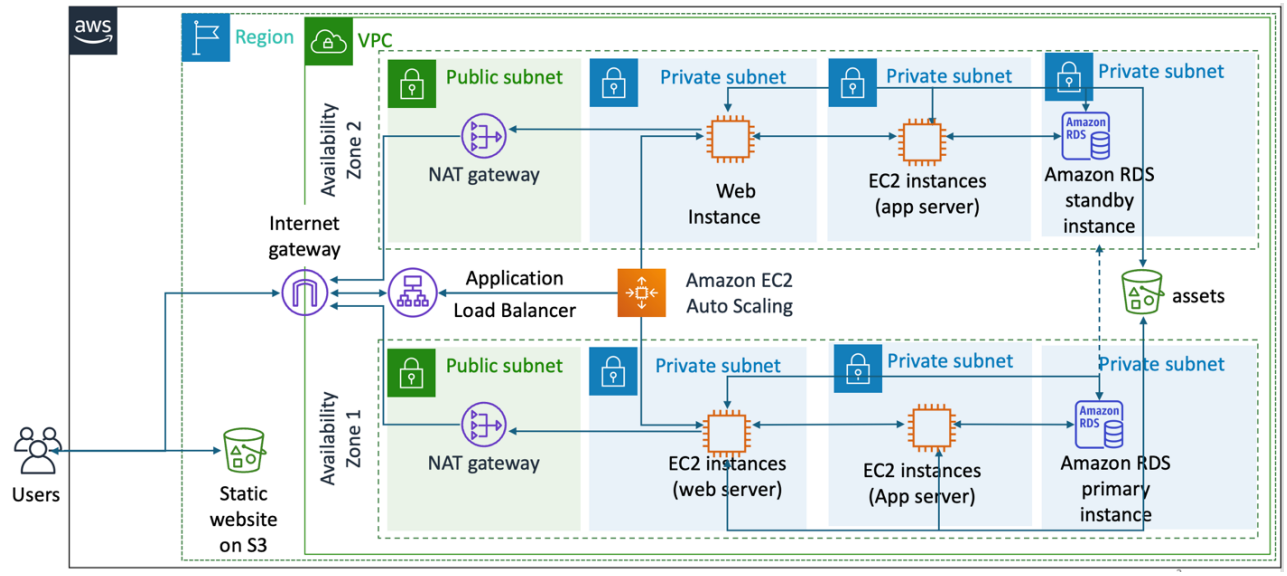
Projet d'Évaluation AWS : Déploiement d'une Application 3 Niveaux
Projet AWS- M1 2024

Objectif

Créer et déployer une architecture complète sur AWS pour une application 3 niveaux (frontend, backend, base de données), puis refactoriser cette architecture en utilisant des conteneurs. Vous appliquerez les meilleures pratiques d'architecture cloud, de sécurité, de haute disponibilité, de scalabilité.



Phase 1 : Infrastructure Réseau

1. **Création du VPC :**
- Configurez un VPC avec un CIDR (ex. : 10.0.0.0/16).
- Créez des sous-réseaux publics et privés dans **au moins deux zones de disponibilité (AZ)** pour assurer la haute disponibilité.

2. **Configuration des Composants Réseau :**
- Créez une **Internet Gateway** pour permettre aux ressources publiques d'accéder à Internet.

- Configurez une ou plusieurs **NAT Gateways** pour permettre aux instances dans les sous-réseaux privés d'accéder à Internet sans être exposées.
- Configurez les tables de routage pour connecter correctement les sous-réseaux publics et privés.

Phase 2 : Sécurité

4. **Groupes de Sécurité :**
- Configurez un groupe de sécurité distinct pour chaque groupe de services (frontend, backend, base de données) avec les règles suivantes :
- **Frontend** : Autorisez uniquement le trafic HTTP/HTTPS depuis Internet via le load balancer.
- **Backend** : Autorisez uniquement les connexions venant du frontend (par exemple, sur le port 8080).
- **Base de Données** : Autorisez uniquement les connexions depuis le backend (par exemple, port 3306 pour MySQL).

5. **Machine Bastion :**
- Ajoutez une instance EC2 bastion dans un sous-réseau public pour se connecter aux machines dans les sous-réseaux privés.
- Limitez l'accès SSH à la machine bastion uniquement depuis une adresse IP spécifique (celle de votre poste de travail).

Phase 3 : Déploiement des Instances

7. **Load Balancer et Auto Scaling :**
- Configurez un **Application Load Balancer (ALB)** pour gérer le trafic du frontend et un autre pour le backend, selon les besoins.
- Mettez en place des groupes d'**Auto Scaling** pour les serveurs web et d'application :
- Définissez des règles de scaling basées sur des métriques comme la CPU Utilization ou le nombre de requêtes.

8. **Base de Données RDS :**
- Configurez une base de données **RDS** (MySQL, PostgreSQL, ou autre) dans des sous-réseaux privés.
- Activez la réplication multi-AZ pour assurer la haute disponibilité.
- Assurez la gestion des sauvegardes automatiques et des snapshots manuels. **(Bonus)**

9. **Stockage des Assets :**
- Déployez un bucket **S3** pour stocker les fichiers statiques et les assets de l'application.

Phase 4 : Refactorisation avec Conteneurs

10. Reproduisez l'architecture EC2 en utilisant des conteneurs :
- Déployez un cluster EKS avec des nœuds de travail (EC2 ou Fargate selon vos besoins).
- Conteneurisez vos applications et services existants en utilisant Docker.
- Créez des manifests Kubernetes pour chaque application/service :
  - Pods : Pour héberger les conteneurs.
  - Deployments : Pour gérer la disponibilité et les mises à jour.
  - Services : Pour exposer les applications (ClusterIP, NodePort ou LoadBalancer).

Phase 5 : Documentation et Présentation

15. **Documentation :**
- Fournissez une documentation complète comprenant les schémas d'architecture, les configurations réseau, les étapes de déploiement.

16. **Présentation :**
- Préparez une présentation finale ou une vidéo de démonstration montrant le déploiement et les fonctionnalités de l'architecture.

Critères d'Évaluation

- Respect des meilleures pratiques AWS (sécurité, haute disponibilité, scalabilité).
- Documentation et clarté des étapes.
- Fonctionnalité et disponibilité de l'application dans les deux versions (EC2 et conteneurs).
- Optimisation des coûts.