Rapport Technique — Projet Capstone AWS

Plan du Rapport — AWS Capstone Project

1. **Introduction :**

1.1 Présentation du projet
1.2 Objectifs techniques
1.3 Technologies et services AWS utilisés

2. **Architecture Globale du Projet :**

2.1 Présentation de l'architecture AWS
2.2 Organisation réseau
2.3 Sécurité de l'infrastructure

3. **Mise en Place du Load Balancer :**

3.1 Création de l'Application Load Balancer
3.2 Création du Target Group
3.3 Vérification du fonctionnement
3.4 Importance du Load Balancer

4. **Déploiement des Instances EC2 :**

4.1 Création du Launch Template
4.2 Déploiement des serveurs applicatifs
4.3 Vérification des instances

5. **Mise en Place de l'Auto Scaling :**

5.1 Création de l'Auto Scaling Group
5.2 Paramètres de capacité
5.3 Fonctionnement de l'Auto Scaling
5.4 Intégration avec le Load Balancer

I. INTRODUCTION

Aujourd'hui, les entreprises modernes dépendent fortement du cloud computing afin d'améliorer la disponibilité de leurs services, leur sécurité, leur flexibilité et leur capacité à évoluer rapidement. Les infrastructures traditionnelles basées sur un seul serveur présentent souvent plusieurs limites importantes : point unique de défaillance, manque de scalabilité, faible tolérance aux pannes et maintenance complexe.

Dans ce projet Capstone, l'entreprise fictive **Example Social Research Organization** souhaitait moderniser son infrastructure applicative. Cette organisation met à disposition des données statistiques mondiales liées à la recherche sociale, telles que :

- La population,
- Le PIB,
- La population urbaine,
- Le nombre d'utilisateurs de téléphones mobiles,
- Le taux de mortalité infantile.

L'application était initialement hébergée dans une architecture classique où le serveur web et la base de données étaient fortement liés et exécutés sur des instances Amazon EC2.

Cette architecture présentait plusieurs problèmes critiques :

- Absence de haute disponibilité,
- Difficulté de montée en charge,
- Faible résilience,
- Risques de sécurité,
- Complexité opérationnelle,
- Exposition potentielle de la base de données.

L'objectif principal du projet était donc de migrer cette infrastructure vers une architecture AWS moderne, sécurisée et hautement disponible utilisant les bonnes pratiques du cloud.

À la fin du projet, l'infrastructure déployée a obtenu un score final de **35/35**, validant le succès complet de l'architecture.

II. Objectifs du Projet

Le projet avait pour objectif principal de transformer l'ancienne infrastructure monolithique en une architecture cloud moderne reposant sur des services AWS managés.

Les objectifs techniques étaient les suivants :

1. Déployer un Application Load Balancer (ALB)
2. Héberger les serveurs applicatifs dans des sous-réseaux privés
3. Mettre en place un Auto Scaling Group
4. Déployer une base de données Amazon RDS MySQL
5. Sécuriser les identifiants avec AWS Secrets Manager
6. Garantir que la base de données reste privée
7. Migrer les données SQL existantes vers RDS
8. Vérifier le bon fonctionnement de l'application
9. Assurer une haute disponibilité multi-zone

III. Présentation du Problème Initial

3.1 Infrastructure Monolithique

Au départ, l'application reposait essentiellement sur une seule instance EC2.

Cela impliquait plusieurs risques :

- Si le serveur tombait en panne, l'application devenait inaccessible ;
- Aucune répartition de charge n'était possible ;
- La maintenance était complexe ;
- Les performances étaient limitées.

3.2 Point Unique de Défaillance :

- L'absence de redondance représentait un problème majeur pour une plateforme utilisée par plusieurs chercheurs.
- Une panne matérielle ou logicielle pouvait interrompre totalement le service.

3.3 Manque de Scalabilité

L'architecture initiale ne permettait pas d'ajuster automatiquement les ressources en fonction du trafic.

En cas d'augmentation du nombre d'utilisateurs :

- Le CPU pouvait saturer ;
- Le site pouvait ralentir ;
- Ou devenir inaccessible.

3.4 Risques de Sécurité

Le stockage manuel des identifiants de base de données dans les fichiers applicatifs représentait un risque important.

Cela pouvait provoquer :

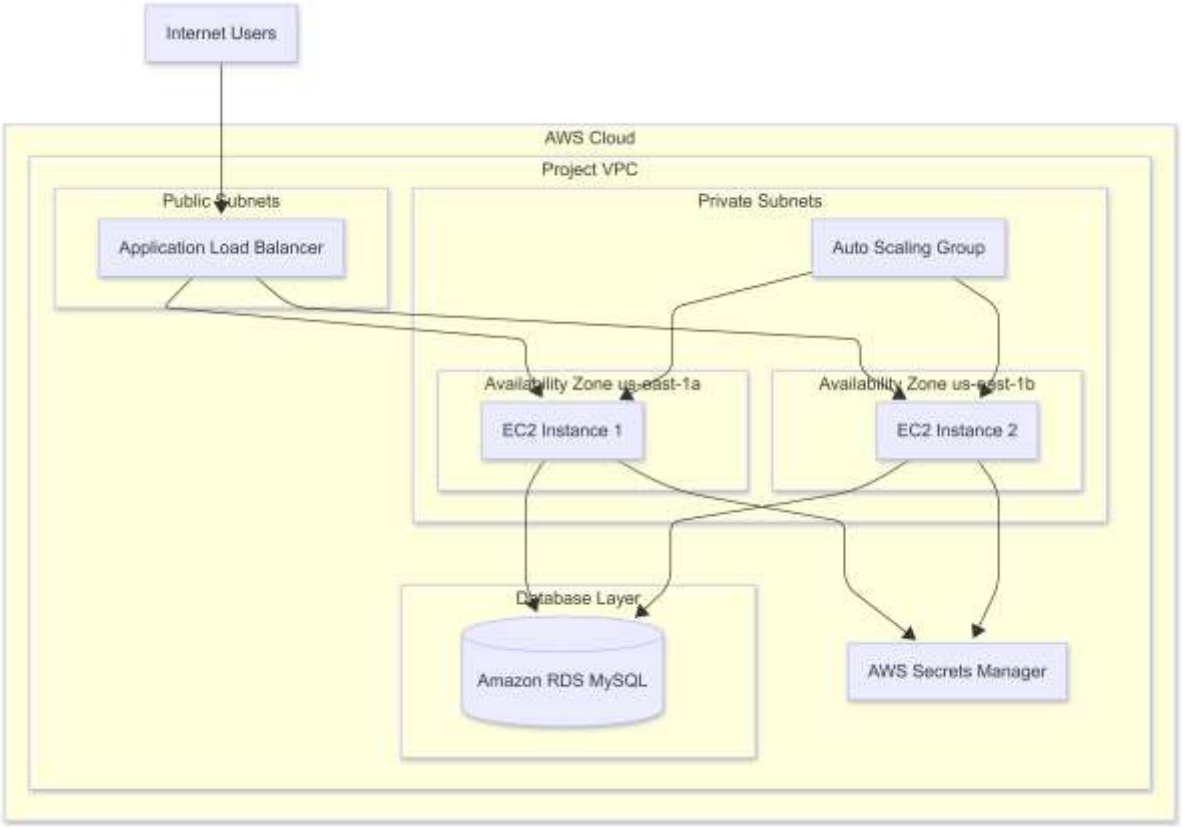
- Une fuite d'identifiants ;
- Un accès non autorisé ;
- Ou une compromission de la base de données.

IV. Architecture Finale de la Solution

La solution mise en place repose sur une architecture cloud moderne à trois couches :

1. **Couche Présentation**
2. **Couche Applicative**
3. **Couche Base de Données**

📌 Schéma Global AWS



V. Amazon VPC — Fondation Réseau

Définition

Amazon VPC (Virtual Private Cloud) est un réseau virtuel isolé dans AWS permettant de déployer des ressources cloud de manière sécurisée.

Le VPC contrôle :

- Les adresses IP,
- Le routage réseau,
- Les sous-réseaux,
- Les règles de sécurité.

Pourquoi le VPC a été utilisé

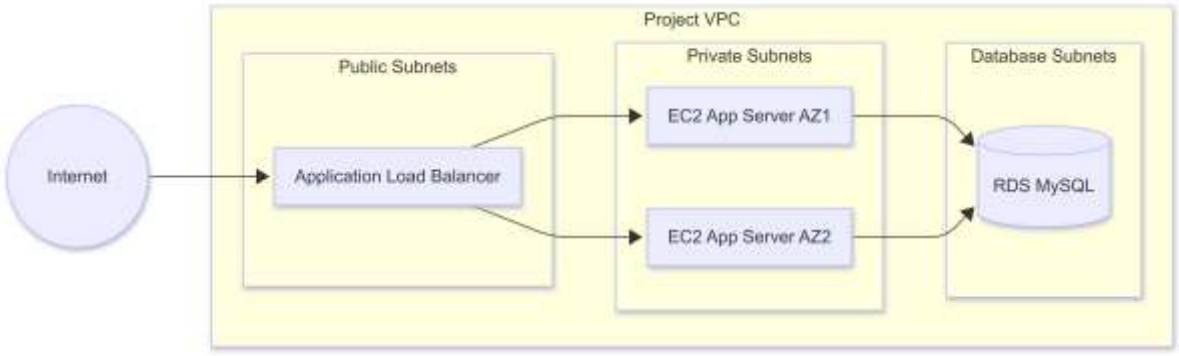
Le VPC permettait :

- D'isoler les ressources ;
- De séparer les composants publics et privés ;
- De contrôler précisément le trafic réseau ;
- D'améliorer la sécurité globale.

L'architecture contenait :

- Des sous-réseaux publics ;
- Des sous-réseaux privés ;
- Plusieurs Availability Zones.

📌 Schéma Réseau VPC



VI. Sous-réseaux Publics et Privés

Sous-réseaux Publics

Les sous-réseaux publics hébergeaient :

- L'Application Load Balancer.

Ces sous-réseaux disposent d'un accès Internet via une Internet Gateway.

Le Load Balancer devait être public afin de recevoir les requêtes HTTP des utilisateurs.

Sous-réseaux Privés

Les sous-réseaux privés hébergeaient :

- Les instances EC2 applicatives ;
- La base de données Amazon RDS.

Les ressources privées ne sont pas directement accessibles depuis Internet, ce qui réduit considérablement la surface d'attaque.

VII. Application Load Balancer (ALB)

Définition

Un Application Load Balancer répartit automatiquement le trafic entrant entre plusieurs serveurs applicatifs.

Pourquoi utiliser un ALB

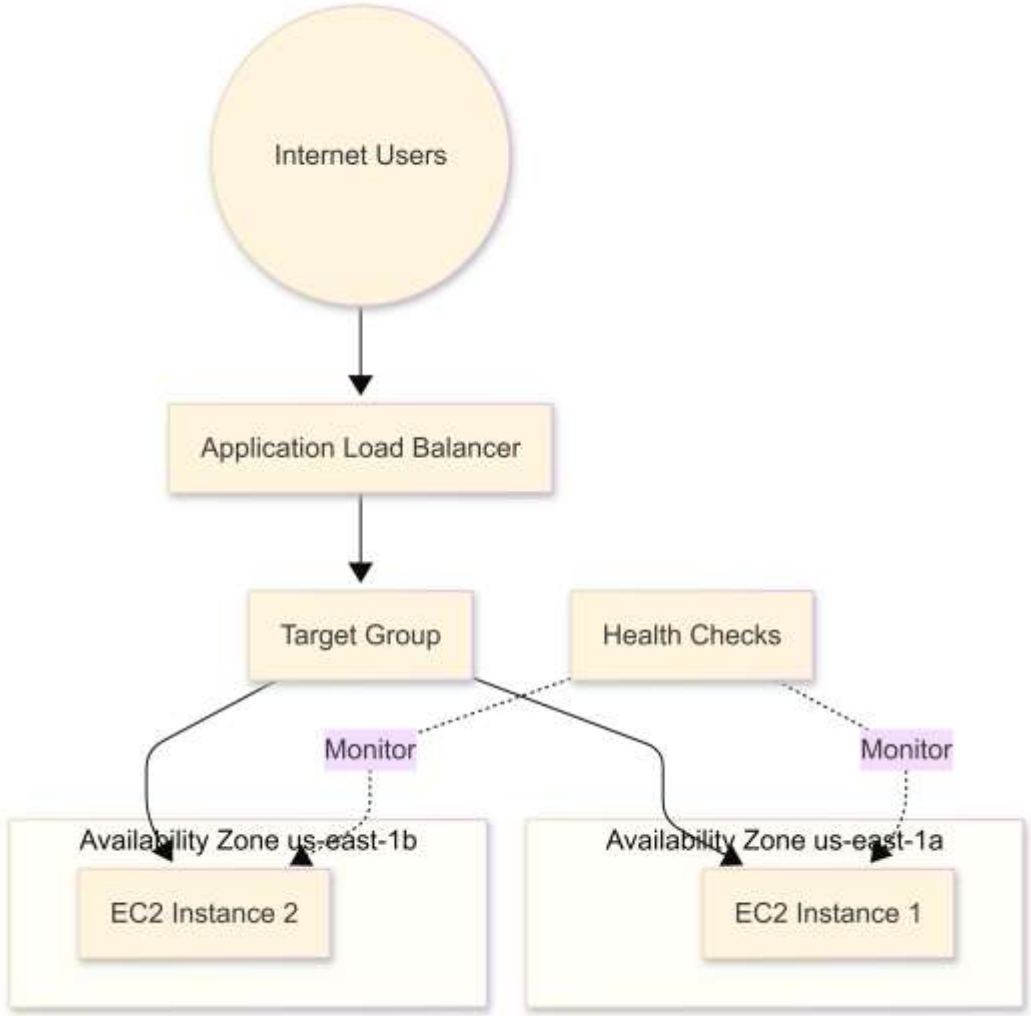
Sans Load Balancer, tout le trafic serait envoyé vers un seul serveur.

L'ALB permet :

- La répartition de charge ;
- L'amélioration de la disponibilité ;
- La tolérance aux pannes ;
- La gestion automatique des serveurs sains.

L’ALB a été déployé dans les sous-réseaux publics et connecté à un groupe cible contenant plusieurs instances EC2.

📌 Schéma Load Balancer



VIII. Target Group et Health Checks

Définition

Le Target Group regroupe les instances EC2 recevant le trafic du Load Balancer.

Health Checks

L'ALB effectue régulièrement des vérifications de santé (health checks).

Si une instance devient indisponible :

- Elle est automatiquement retirée du trafic ;
- Les utilisateurs continuent d'être servis par les autres instances.

IX. Instances Amazon EC2

Définition

Amazon EC2 fournit des serveurs virtuels dans le cloud AWS.

Les instances EC2 hébergeaient :

- Le serveur Apache ;
- L'application PHP ;
- La logique métier.

Pourquoi EC2 a été utilisé

EC2 offre :

- Flexibilité ;
- Puissance de calcul ;
- Contrôle du système ;
- Compatibilité avec les applications Linux classiques.

Les instances ont été déployées dans des sous-réseaux privés pour renforcer la sécurité.

X. Auto Scaling Group (ASG)

Définition

Un Auto Scaling Group gère automatiquement un groupe d'instances EC2.

Il peut :

- Lancer de nouvelles instances ;
- Remplacer les serveurs défaillants ;
- Maintenir une capacité minimale.

Pourquoi utiliser un ASG

L'entreprise souhaitait garantir la haute disponibilité de l'application.

L'ASG permet :

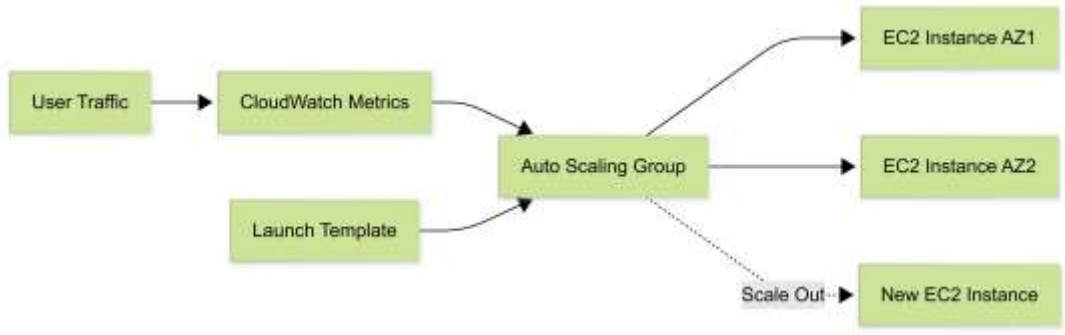
- Le remplacement automatique des instances défaillantes ;
- La répartition multi-zone ;
- L'augmentation automatique du nombre de serveurs si nécessaire.

Deux Availability Zones ont été utilisées :

- us-east-1a
- us-east-1b

Cette approche améliore fortement la résilience de l'infrastructure.

📌 Schéma Auto Scaling



XI. Amazon RDS MySQL

Définition

Amazon RDS est un service managé permettant d'exécuter des bases de données relationnelles sans gérer directement les serveurs.

Pourquoi utiliser RDS

Gérer une base de données manuellement sur EC2 est complexe.

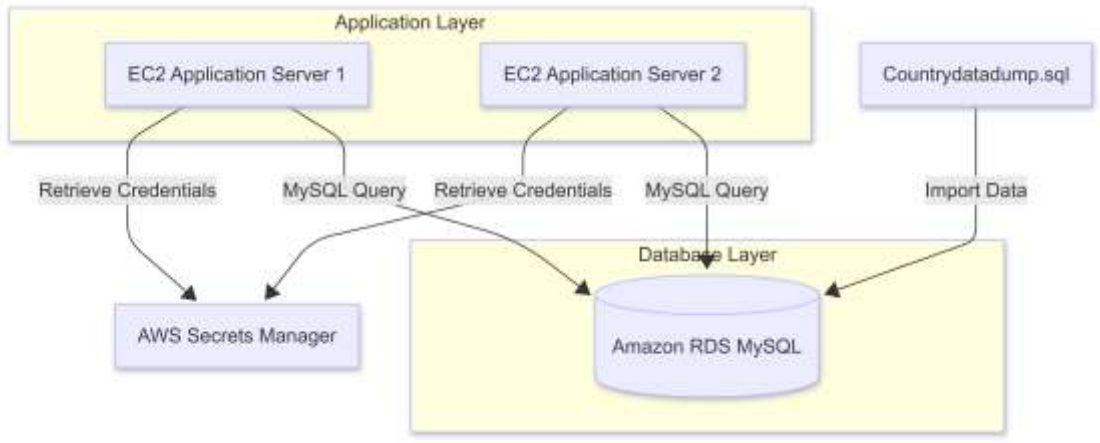
RDS simplifie :

- Les sauvegardes ;
- Les mises à jour ;
- La maintenance ;
- Le monitoring ;
- La sécurité.

Le projet utilisait :

- MySQL Community Edition ;
- Une instance db.t3.micro ;
- Un accès privé uniquement.

📌 Schéma Base de Données :



XII. AWS Secrets Manager

Définition

AWS Secrets Manager permet de stocker et gérer de manière sécurisée :

- Les mots de passe ;
- Les clés API ;
- Les identifiants de base de données.

Pourquoi utiliser Secrets Manager

Le stockage manuel des mots de passe est dangereux.

Secrets Manager améliore :

- La sécurité ;
- La centralisation des secrets ;
- La rotation des identifiants ;
- La protection contre les fuites.

Les instances EC2 récupéraient dynamiquement les identifiants de connexion à la base RDS.

XIII. Groupes de Sécurité (Security Groups)

Définition

Les Security Groups agissent comme des pare-feux virtuels contrôlant le trafic entrant et sortant.

Architecture Sécurisée

Security Group du Load Balancer

Autorise :

- HTTP (port 80) depuis Internet.

Security Group des EC2

Autorise :

- HTTP uniquement depuis l'ALB.

Security Group de RDS

Autorise :

- MySQL (port 3306) uniquement depuis les instances EC2.

Cette approche limite fortement les risques d'intrusion.

XIV. AWS Systems Manager Session Manager

Définition

Session Manager permet d'accéder aux instances EC2 sans SSH public ni clé SSH.

Pourquoi Session Manager a été utilisé

Cette solution :

- Évite l'exposition des ports SSH ;
- Simplifie l'administration ;
- Améliore la sécurité ;
- Réduit les risques d'attaque.

Les connexions étaient entièrement gérées via AWS Systems Manager.

XV. Migration des Données SQL

Une étape importante du projet consistait à migrer les données existantes vers Amazon RDS.

Téléchargement du fichier SQL

Le dump SQL a été téléchargé depuis Amazon S3.

Commande utilisée :

wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACACAD-3-113230/22-lab-Capstone-project/s3/Countrydatadump.sql

Importation dans Amazon RDS

Les données ont ensuite été importées dans MySQL RDS.

Commande utilisée :

mysql -h <RDS-ENDPOINT> -u admin -p countries < Countrydatadump.sql

XVI. Validation de l'Application

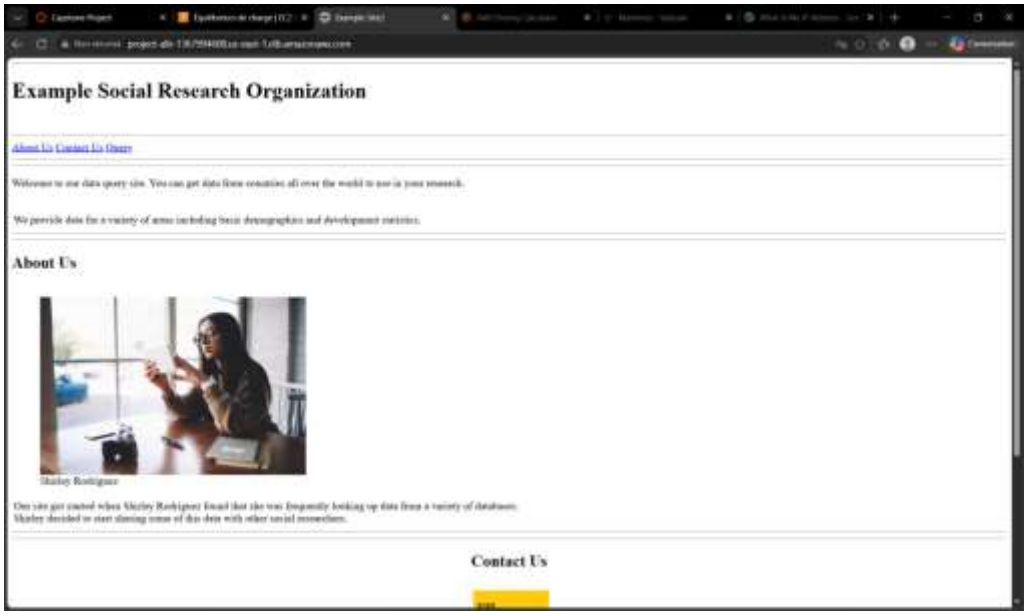
Après la migration :

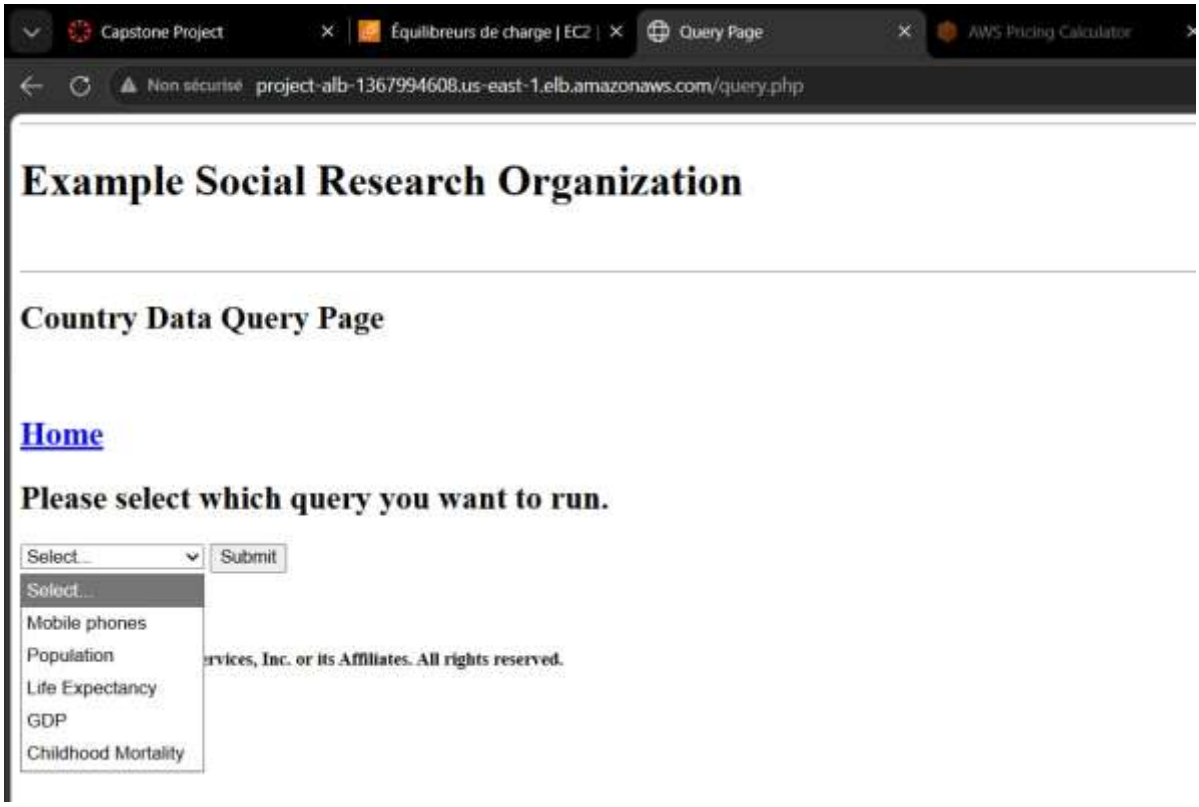
- Toutes les requêtes SQL fonctionnaient correctement ;
- Les pages web affichaient les données attendues ;
- L'ALB distribuait correctement le trafic.

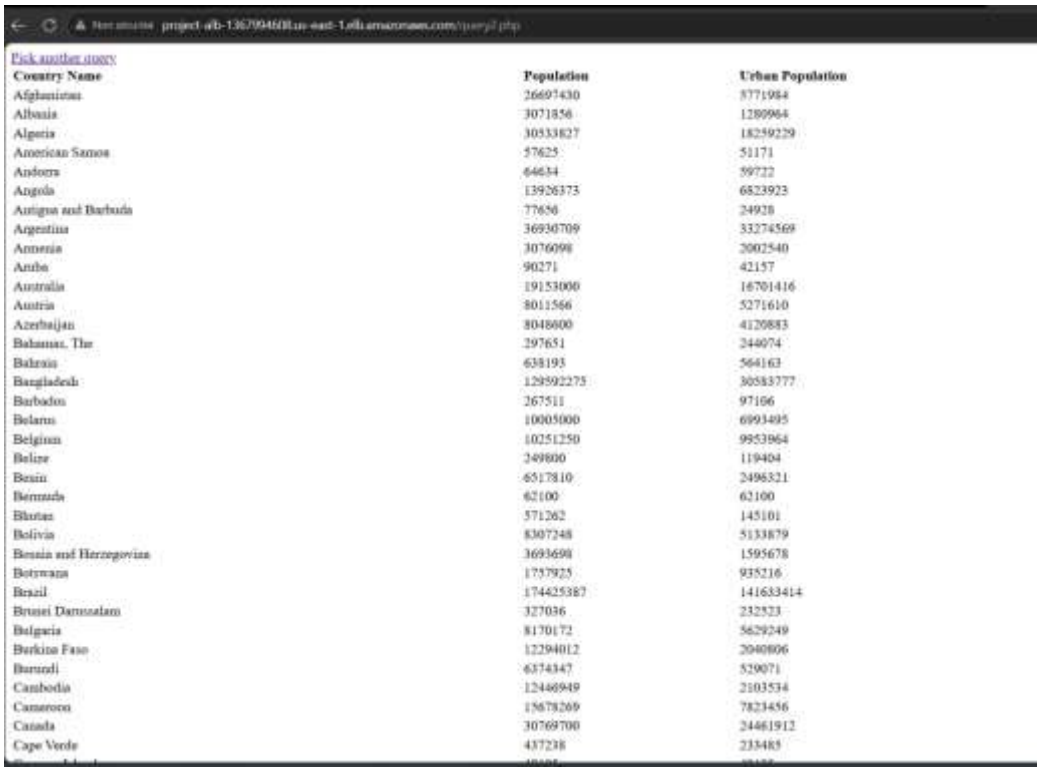
Les requêtes disponibles incluaient :

- Population
- PIB
- Population urbaine
- Téléphones mobiles
- Mortalité infantile

📌 Captures d'Écran

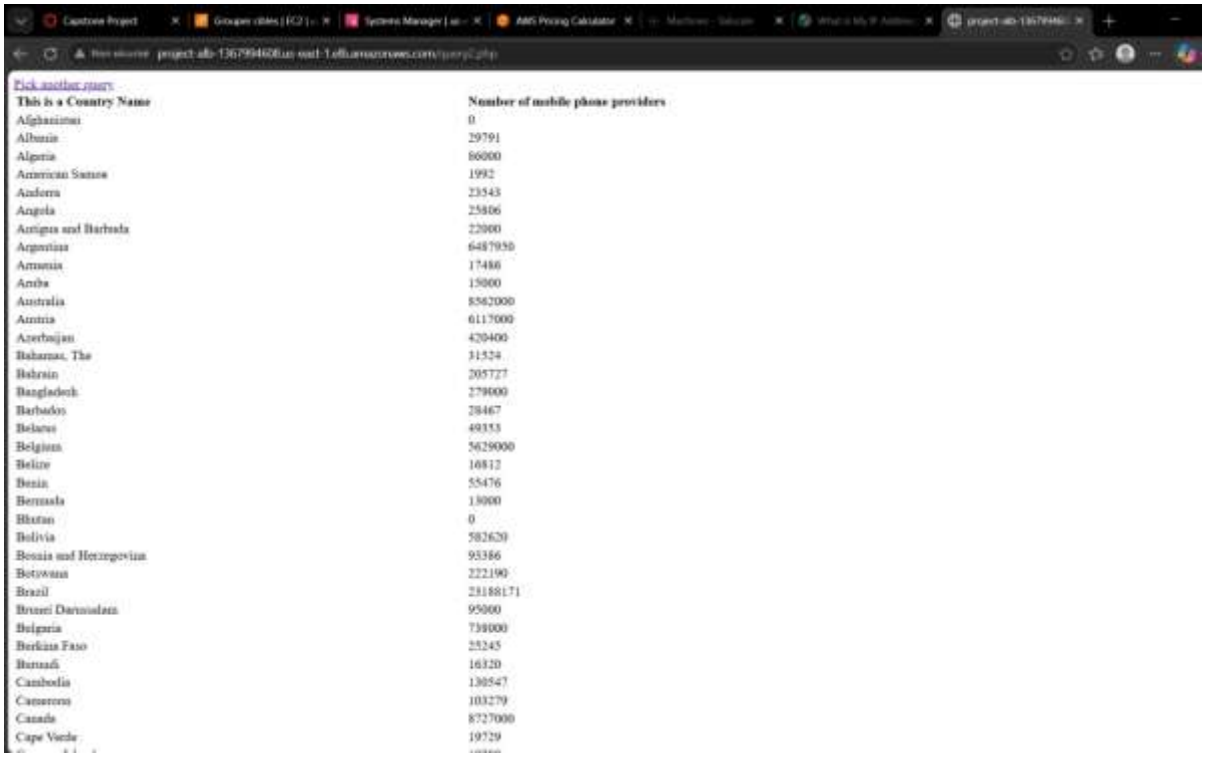






Pick another query

Country Name	Population	Urban Population
Afghanistan	26697430	5771984
Albania	3071856	1280964
Algeria	30533827	18259229
American Samoa	57625	51171
Andorra	64634	59722
Angola	13926373	6823923
Antigua and Barbuda	77656	24928
Argentina	36930709	33274569
Armenia	3076098	2002540
Aruba	90271	42157
Australia	19153000	16701416
Austria	8011566	5271610
Azerbaijan	8048600	4120883
Bahamas, The	297651	244074
Bahrain	638193	564163
Bangladesh	129592275	30583777
Barbados	267511	97106
Belarus	10005000	6993495
Belgium	10251250	9953964
Belize	249800	119404
Benin	6517810	2496321
Bermuda	62100	62100
Bhutan	571262	145101
Bolivia	8307248	5133879
Bosnia and Herzegovina	3693698	1595678
Botswana	1757925	935216
Brazil	174425387	141633414
Brunei Darussalam	327036	232523
Bulgaria	8170172	5629249
Burkina Faso	12294012	2040806
Burundi	6374347	529071
Cambodia	12446949	2103534
Cameroon	15678269	7823456
Canada	30769700	24461912
Cape Verde	437238	233485



Pick another query

This is a Country Name	Number of mobile phone providers
Afghanistan	0
Albania	29791
Algeria	86000
American Samoa	1992
Andorra	23543
Angola	25806
Antigua and Barbuda	22000
Argentina	6487950
Armenia	17486
Aruba	15000
Australia	8562000
Austria	6117000
Azerbaijan	420400
Bahamas, The	31524
Bahrain	205727
Bangladesh	279000
Barbados	28467
Belarus	49353
Belgium	5629000
Belize	16812
Benin	55476
Bermuda	13000
Bhutan	0
Bolivia	582620
Bosnia and Herzegovina	95386
Botswana	222190
Brazil	28188171
Brunei Darussalam	95000
Bulgaria	738000
Burkina Faso	25245
Burundi	16320
Cambodia	130547
Cameroon	103279
Canada	8727000
Cape Verde	19729

XVII. Haute Disponibilité et Tolérance aux Pannes

L'architecture finale garantit une forte résilience grâce à :

- L'Application Load Balancer ;
- L'Auto Scaling Group ;
- Le déploiement multi-AZ ;
- Les health checks automatiques.

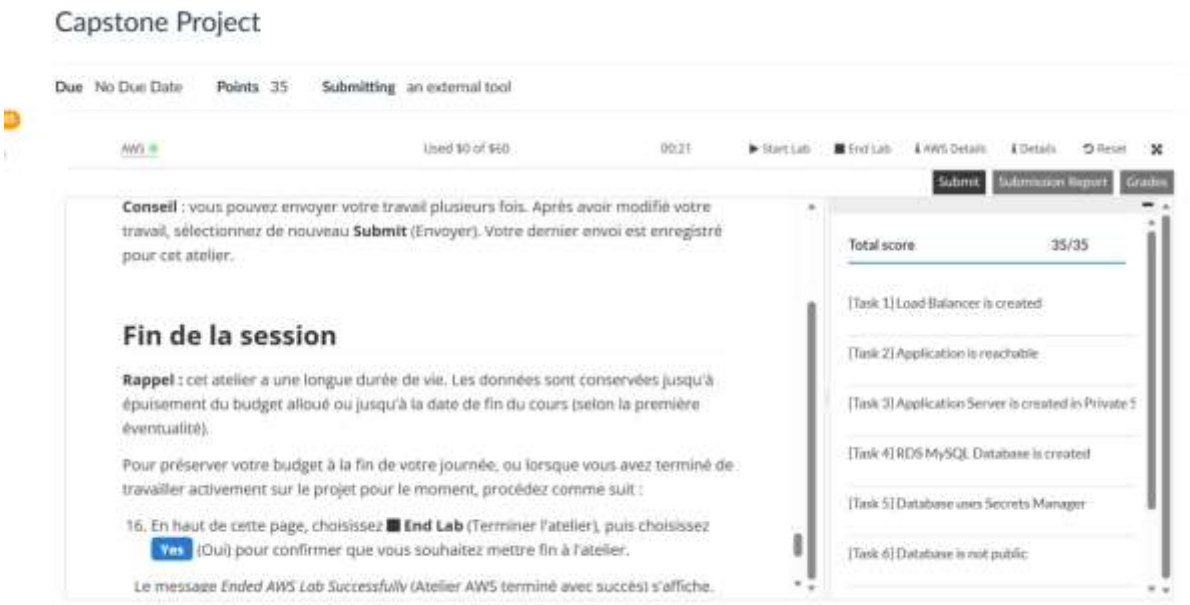
En cas de panne :

1. **L'ALB cesse d'envoyer du trafic vers l'instance défaillante ;**
2. **L'ASG recrée automatiquement une nouvelle instance ;**

L'application reste accessible.

XVIII. Résultats Finaux

Le projet a été validé avec succès.



XIX. Conclusion

Ce projet Capstone a permis de transformer une infrastructure classique vulnérable en une architecture cloud moderne, sécurisée et hautement disponible sur AWS.

L'utilisation combinée de :

- VPC,
- Load Balancer,
- Auto Scaling,
- Amazon RDS,
- Secrets Manager et Systems Manager a permis de construire une solution professionnelle répondant aux bonnes pratiques AWS.

L'infrastructure finale garantit :

- Une meilleure sécurité ;
- Une haute disponibilité ;
- Une forte résilience ;
- Une maintenance simplifiée et une meilleure expérience utilisateur.

Le projet a atteint tous les objectifs fixés et obtenu la note maximale de **35/35**, validant le succès complet de l'implémentation cloud.