

# Faculté des sciences Tunis

## Projet AWS :

*Réalisé par :*

*Mohamed Habib Manai*

*Slouma Rayen*

*Classe : IGL4*

*Décembre 2024*

Architecture Cloud AWS  
Application Web à Trois Niveaux

# Table des matières

Introduction . . . . .	2
0.1 Étape 1 : Configuration du VPC . . . . .	2
0.1.1 1.1 Configuration Principale . . . . .	2
0.1.2 1.2 Configuration des Sous-réseaux . . . . .	2
0.1.3 1.3 Tables de Routage . . . . .	2
0.1.4 1.4 Connexions Réseau . . . . .	3
0.2 Étape 2 : Configuration des Groupes de Sécurité . . . . .	4
0.2.1 2.1 Vue d'Ensemble des Groupes de Sécurité . . . . .	4
0.2.2 2.2 Configuration Détaillée des Groupes de Sécurité . . . . .	4
0.3 Étape 3 : Configuration de la Base de Données RDS . . . . .	7
0.3.1 3.1 Configuration du Groupe de Sous-réseaux RDS . . . . .	7
0.3.2 3.2 Configuration de l'Instance RDS . . . . .	7
0.4 Étape 4 : Déploiement et Configuration des Instances . . . . .	8
0.4.1 4.1 Instance Bastion . . . . .	8
0.4.2 4.2 Configuration Backend . . . . .	9
0.4.3 4.3 Configuration Frontend . . . . .	9
0.5 Étape 5 : Configuration des Load Balancers . . . . .	10
0.5.1 5.1 Load Balancer Public (Frontend) . . . . .	10
0.5.2 5.2 Load Balancer Interne (Backend) . . . . .	10
0.6 Étape 6 : Configuration de l'Auto Scaling . . . . .	11
0.6.1 6.1 Auto Scaling Group Frontend . . . . .	11
0.6.2 6.2 Auto Scaling Group Backend . . . . .	12
0.7 Étape 7 : Configuration des Target Groups . . . . .	12
0.7.1 7.1 Frontend Target Group . . . . .	12
0.7.2 7.2 Backend Target Group . . . . .	12
0.8 Conclusion . . . . .	13
Conclusion . . . . .	13

## Introduction

Ce projet consiste en la mise en place d'une architecture cloud complète sur AWS pour une application web à trois niveaux. L'objectif est de créer une infrastructure hautement disponible, sécurisée et évolutive, en suivant les meilleures pratiques d'AWS.

### 0.1 Étape 1 : Configuration du VPC

#### 0.1.1 1.1 Configuration Principale

```
Nom: projet_vpc
ID: vpc-0c8323d079a5de71a
CIDR IPv4: 192.0.0.0/16
État: Available
Configuration DNS:
  Résolution DNS: Enabled
  DNS hostnames: Disabled
Options:
  Tenancy: default
  Default VPC: No
  Block Public Access: Off
  Network Address Usage metrics: Disabled
```

#### 0.1.2 1.2 Configuration des Sous-réseaux

```
Zone us-east-1a:
- public_subnet_AZ1:
  CIDR: 192.0.0.0/24
- private_web_frontend_subnet_AZ1:
  CIDR: 192.0.2.0/24
- private_web_backend_subnet_AZ1:
  CIDR: 192.0.4.0/24
- private_web_RDS_subnet_AZ1:
  CIDR: 192.0.6.0/24

Zone us-east-1b:
- public_subnet_AZ2:
  CIDR: 192.0.1.0/24
- private_web_frontend_subnet_AZ2:
  CIDR: 192.0.3.0/24
- private_web_backend_subnet_AZ2:
  CIDR: 192.0.5.0/24
- private_web_RDS_subnet_AZ2:
  CIDR: 192.0.7.0/24
```

#### 0.1.3 1.3 Tables de Routage

```
Tables de Routage:
- Main Route Table:
  ID: rtb-0ddd8647def85bbce
  Associations: Aucune association de sous-réseau
  Routes: 1 route incluant local
```

- PRIVATE\_RT\_AZ1:
  - Associations: 3 sous-réseaux
  - Routes:
    - Destination: 0.0.0.0/0  
Target: nat-0334bea99f2385c6f (NAT\_AZ1)
    - Destination: 192.0.0.0/16  
Target: local
  - Sous-réseaux associés:
    - private\_web\_RDS\_subnet\_AZ1
    - private\_web\_frontend\_subnet\_AZ1
    - private\_web\_backend\_subnet\_AZ1
- PRIVATE\_RT\_AZ2:
  - Associations: 3 sous-réseaux
  - Routes:
    - Destination: 0.0.0.0/0  
Target: nat-09876543210 (NAT\_AZ2)
    - Destination: 192.0.0.0/16  
Target: local
  - Sous-réseaux associés:
    - private\_web\_RDS\_subnet\_AZ2
    - private\_web\_frontend\_subnet\_AZ2
    - private\_web\_backend\_subnet\_AZ2
- PUBLIC\_RT:
  - ID: rtb-04a054ca9a79737fc
  - Routes:
    - Destination: 0.0.0.0/0  
Target: igw-001891f6cbf1da577 (projet\_IGW)
    - Destination: 192.0.0.0/16  
Target: local
  - Sous-réseaux associés:
    - public\_subnet\_AZ2
    - public\_subnet\_AZ1

#### 0.1.4 1.4 Connexions Réseau

##### Connexions:

- Internet Gateway:
  - ID: igw-001891f6cbf1da577
  - Nom: projet\_IGW
  - État: Attached
  - VPC: vpc-0c8323d079a5de71a
- NAT Gateways:
  - NAT\_AZ1:
    - Type: NAT Gateway public
    - Configuration: 1 ENI avec 1 EIP
    - État: Available
    - Adresse IPv4 Publique: 35.172.132.32
    - Adresse IPv4 Privée: 192.0.0.223

NAT\_AZ2:

Type: NAT Gateway public

Configuration: 1 ENI avec 1 EIP

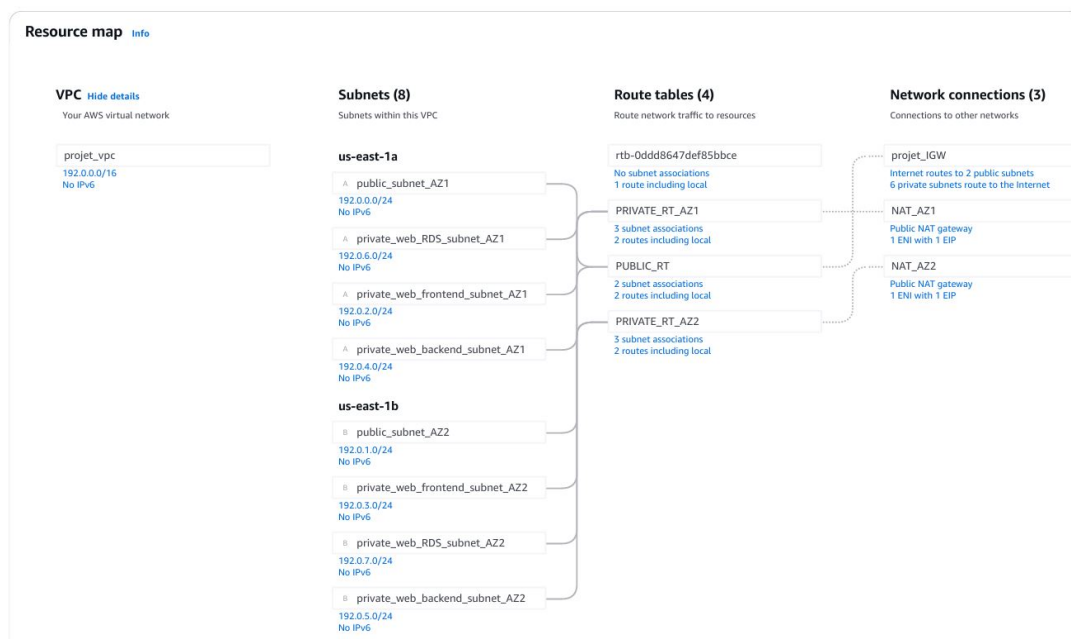


FIGURE 1 – Configuration VPC.

#### Architecture VPC

- Architecture multi-AZ avec répartition dans deux zones de disponibilité
- Séparation claire entre sous-réseaux publics et privés
- Isolation des couches applicatives (frontend, backend, RDS) dans des sous-réseaux dédiés
- Haute disponibilité avec NAT Gateways redondants dans chaque AZ
- Sécurisation avec contrôle d'accès via Network ACLs et tables de routage

## 0.2 Étape 2 : Configuration des Groupes de Sécurité

### 0.2.1 2.1 Vue d'Ensemble des Groupes de Sécurité

Groupes de Sécurité Configurés:

- **internal\_lb\_sg**: Groupe de sécurité pour le load balancer interne
- **backend\_sg**: Groupe de sécurité pour les instances backend
- **RDS\_sg**: Groupe de sécurité pour la base de données
- **Bastion\_sg**: Groupe de sécurité pour l'instance bastion
- **frontend\_sg**: Groupe de sécurité pour les instances frontend
- **front\_lb\_sg**: Groupe de sécurité pour le load balancer frontend

### 0.2.2 2.2 Configuration Détaillée des Groupes de Sécurité

Frontend Security Group (frontend\_sg)

Security Group ID: sg-09e7a872e577b8c89

Description: frontend\_sg

**Règles Entrantes:**

- Type: HTTP  
Port: 80  
Source: front\_lb\_sg  
Protocol: TCP
- Type: SSH  
Port: 22  
Source: Bastion\_sg  
Protocol: TCP

**Règles Sortantes:**

- Type: All traffic  
Protocol: All  
Destination: 0.0.0.0/0

**Backend Security Group (backend\_sg)**

Security Group ID: sg-03019fea108ab8e7d

Description: backend\_sg

**Règles Entrantes:**

- Type: SSH  
Port: 22  
Source: Bastion\_sg  
Protocol: TCP
- Type: Custom TCP  
Port: 4000  
Source: internal\_lb\_sg  
Protocol: TCP

**Règles Sortantes:**

- Type: All traffic  
Protocol: All  
Destination: 0.0.0.0/0

**RDS Security Group (RDS\_sg)**

Security Group ID: sg-080ca26023114968b

Description: RDS\_sg

**Règles Entrantes:**

- Type: MYSQL/Aurora  
Port: 3306  
Source: backend\_sg  
Protocol: TCP

**Règles Sortantes:**

- Type: All traffic  
Protocol: All  
Destination: 0.0.0.0/0

**Load Balancer Security Groups**

Front Load Balancer (front\_lb\_sg):

ID: sg-03fbaf6a30cad6c59

**Règles Entrantes:**

- Type: HTTP  
Port: 80  
Source: 0.0.0.0/0  
Protocol: TCP
- Type: HTTP  
Port: 80  
Source: ::/0  
Protocol: TCP (IPv6)

**Internal Load Balancer (internal\_lb\_sg):**

ID: sg-0e68890e7c0c3b392

**Règles Entrantes:**

- Type: HTTP  
Port: 80  
Source: frontend\_sg  
Protocol: TCP

**Règles Sortantes (Both):**

- Type: All Traffic  
Destination: 0.0.0.0/0

**Bastion Security Group (Bastion\_sg)**

Security Group ID: sg-000be5e9080881a8c

Description: allows ssh

**Règles Entrantes:**

- Type: SSH  
Port: 22  
Source: 197.2.174.9/32  
Protocol: TCP

**Règles Sortantes:**

- Type: All traffic  
Protocol: All  
Destination: 0.0.0.0/0

Security Groups (8) <small>Info</small>							
<input type="text" value="Find resources by attribute or tag"/>				<a href="#">Actions</a>	<a href="#">Export security groups to CSV</a>	<a href="#">Create security group</a>	
<input type="checkbox"/>	Name	Security group ID	Security group name	VPC ID	Description		
<input type="checkbox"/>	-	<a href="#">sg-04990cebf0dae997</a>	default	<a href="#">vpc-0e14fe556cdd79639</a>	default VPC security group		
<input type="checkbox"/>	-	<a href="#">sg-0e68890e7c0c3b392</a>	internal_lb_sg	<a href="#">vpc-0c8323d079a5de71a</a>	internal_lb_sg		
<input type="checkbox"/>	-	<a href="#">sg-03019fca108ab8e7d</a>	backend_sg	<a href="#">vpc-0c8323d079a5de71a</a>	backend_sg		
<input type="checkbox"/>	-	<a href="#">sg-080ca26023114968b</a>	RDS_sg	<a href="#">vpc-0c8323d079a5de71a</a>	RDS_sg		
<input type="checkbox"/>	-	<a href="#">sg-000be5e9080881a8c</a>	Bastion_sg	<a href="#">vpc-0c8323d079a5de71a</a>	allows ssh		
<input type="checkbox"/>	-	<a href="#">sg-09e7a872e577b8c89</a>	frontend_sg	<a href="#">vpc-0c8323d079a5de71a</a>	frontend_sg		
<input type="checkbox"/>	-	<a href="#">sg-0cf3990f89373441c</a>	default	<a href="#">vpc-0c8323d079a5de71a</a>	default VPC security group		
<input type="checkbox"/>	-	<a href="#">sg-03fba6a30cad6c59</a>	front_lb_sg	<a href="#">vpc-0c8323d079a5de71a</a>	front_lb_sg		

FIGURE 2 – Security Groups.

#### Points Clés de Sécurité

- Isolation complète des composants avec des groupes de sécurité dédiés
- Accès SSH limité via le bastion host avec une IP source spécifique
- Communication entre les couches strictement contrôlée
- Base de données accessible uniquement depuis le backend
- Load balancers configurés pour leur rôle spécifique (public/interne)

## 0.3 Étape 3 : Configuration de la Base de Données RDS

### 0.3.1 3.1 Configuration du Groupe de Sous-réseaux RDS

VPC ID: vpc-0c8323d079a5de71a

ARN: arn:aws:rds:us-east-1:436156172735:subgrp/rds\_subnet\_group

Type de Réseau Supporté: IPv4

Description: RDS\_subnet\_group

Sous-réseaux:

Zone us-east-1a:

- Nom: private\_web\_RDS\_subnet\_AZ1
- ID: subnet-0688ad1effeff9c99
- CIDR: 192.0.6.0/24

Zone us-east-1b:

- Nom: private\_web\_RDS\_subnet\_AZ2
- ID: subnet-089b2ade119801248
- CIDR: 192.0.7.0/24

### 0.3.2 3.2 Configuration de l'Instance RDS

DB Identifier: projetdb

Engine: MySQL Community (8.0.39)

Instance Class: db.t3.micro

Statut: Available

Storage: 20 GiB (GP2)

Configuration:

- Multi-AZ: Yes
- Zone Secondaire: us-east-1a
- vCPU: 2
- RAM: 1 GB
- Endpoint: projetdb.cvbzdkdjny.us-east-1.rds.amazonaws.com
- Port: 3306

Paramètres Avancés:

- Master Username: admin
- RDS Extended Support: Disabled
- Storage Autoscaling: Enabled
- Max Storage Threshold: 1000 GiB



**projetdb** Modify Actions

Summary				
DB identifier projetdb	Status Available	Role Instance	Engine MySQL Community	Recommendations 1 informational
CPU 4.67%	Class db.t3.micro	Current activity 2 Connections	Region & AZ us-east-1b	

Connectivity & security | Monitoring | Logs & events | Configuration | Zero-ETL integrations | Maintenance & backups | Data migrations - new | Tags | Recommendations

**Connectivity & security**

**Endpoint & port**

Endpoint  
projetdb.cv3ztdjdqyr.us-east-1.rds.amazonaws.com

Port  
3306

**Networking**

Availability Zone  
us-east-1b

VPC  
projec\_vpc (vpc-0c8323d079a5de71a)

Subnet group  
rds\_subnet\_group

Subnets  
subnet-0588ad1eff9c99  
subnet-0d962e9d1198012c8

Network type  
IPv4

**Security**

VPC security groups  
RDS\_sg (sg-080ca26023114968b)  
Active

Publicly accessible  
No

Certificate authority  
rds-ca-rsa2048-g1

Certificate authority date  
May 26, 2061, 00:34 (UTC+01:00)

DB instance certificate expiration date  
December 22, 2025, 07:16 (UTC+01:00)

FIGURE 3 – RDS Database.

## 0.4 Étape 4 : Déploiement et Configuration des Instances

### 0.4.1 4.1 Instance Bastion

Instance ID: i-0b2061cf2532271a1

Type: t2.micro

Public IP: 34.200.23.62

Private IP: 192.0.0.8

Subnet: public\_subnet\_AZ1

État: Running

**Instance summary for i-0b2061cf2532271a1 (bastion\_host)** Info Connect Instance state Actions

Updated less than a minute ago

<b>Instance ID</b> i-0b2061cf2532271a1	<b>Public IPv4 address</b> 34.200.23.62   <a href="#">open address</a>	<b>Private IPv4 addresses</b> 192.0.0.8
<b>IPv6 address</b> -	<b>Instance state</b> Running	<b>Public IPv4 DNS</b> -
<b>Hostname type</b> IP name: ip-192-0-0-8.ec2.internal	<b>Private IP DNS name (IPv4 only)</b> ip-192-0-0-8.ec2.internal	<b>Elastic IP addresses</b> 34.200.23.62 [Public IP]
<b>Answer private resource DNS name</b> -	<b>Instance type</b> t2.micro	<b>AWS Compute Optimizer finding</b> Opt-in to AWS Compute Optimizer for recommendations.   <a href="#">Learn more</a>
<b>Auto-assigned IP address</b> -	<b>VPC ID</b> vpc-0c8323d079a5de71a (projec_vpc)	<b>Auto Scaling Group name</b> -
<b>IAM Role</b> -	<b>Subnet ID</b> subnet-0ab071f010225baf6 (public_subnet_AZ1)	<b>Managed</b> false
<b>IMDSv2</b> Required	<b>Instance ARN</b> arn:aws:ec2:us-east-1:436156172735:instance/i-0b2061cf2532271a1	
<b>Operator</b> -		

FIGURE 4 – Bastion instance.

## 0.4.2 4.2 Configuration Backend

```
# Installation de MySQL Client
sudo yum install -y https://dev.mysql.com/get/mysql80-community-release-el9-5.noarch.rpm
sudo yum install -y mysql-community-server
sudo systemctl enable --now mysqld

# Configuration Node.js
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
source ~/.bashrc
nvm install 16
nvm use 16

# Installation des dépendances
npm install -g pm2
sudo yum install -y git
git clone https://github.com/Rayen-Slouma/aws_project.git
cd /home/ec2-user/aws_project/Notes-web-app/application-code/app-tier
npm install mysql express body-parser
npm install
npm audit fix

# Démarrage avec PM2
pm2 start index.js
sudo env PATH=$PATH:/home/ec2-user/.nvm/versions/node/v16.20.0/bin \
/home/ec2-user/.nvm/versions/node/v16.20.0/lib/node_modules/pm2/bin/pm2 \
startup systemd -u ec2-user --hp /home/ec2-user

# Configuration Base de Données
mysql -h projetdb.cv3zdlldjrqr.us-east-1.rds.amazonaws.com -u admin -p
```

## 0.4.3 4.3 Configuration Frontend

```
# Installation Node.js
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
source ~/.bashrc
nvm install 16
nvm use 16

# Installation des dépendances
sudo yum install -y git
git clone https://github.com/Rayen-Slouma/aws_project.git
cd /home/ec2-user/aws_project/Notes-web-app/application-code/web-tier
npm install
npm audit fix
chmod +x /home/ec2-user/aws_project/Notes-web-app/application-code/web-tier/node_modules/.bin
npm run build

# Configuration Nginx
sudo yum install nginx
sudo cp /home/ec2-user/aws_project/Notes-web-app/application-code/nginx.conf /etc/nginx/
chmod -R 755 /home/ec2-user
```

```
sudo chkconfig nginx on
```

## 0.5 Étape 5 : Configuration des Load Balancers

### 0.5.1 5.1 Load Balancer Public (Frontend)

Nom: publiclb

Type: Application

Scheme: Internet-facing

État: Active

Zone hébergée: Z355XD0TRQ7X7K

VPC: vpc-0c8323d079a5de71a

Availability Zones:

- us-east-1b (use1-az4)
- us-east-1a (use1-az2)

ARN: arn:aws:elasticloadbalancing:us-east-1:436156172735:loadbalancer/app/publiclb/ba38a85c52

DNS: publiclb-2138926821.us-east-1.elb.amazonaws.com

Listener Configuration:

Protocol: HTTP

Port: 80

Action par défaut: Forward to frontendservertargetgroup

Target Group Stickiness: Off

#### publiclb



##### ▼ Details

###### Load balancer type

Application

###### Status

Active

###### VPC

[vpc-0c8323d079a5de71a](#)

###### Load balancer IP address type

IPv4

###### Scheme

Internet-facing

###### Hosted zone

Z355XD0TRQ7X7K

###### Availability Zones

[subnet-0aaef77439e4d3a48](#) us-east-1b (use1-az4)  
[subnet-0ab071f010225baf6](#) us-east-1a (use1-az2)

###### Date created

December 22, 2024, 14:10 (UTC+01:00)

###### Load balancer ARN

[arn:aws:elasticloadbalancing:us-east-1:436156172735:loadbalancer/app/publiclb/ba38a85c52a3decf](#)

###### DNS name Info

[publiclb-2138926821.us-east-1.elb.amazonaws.com](#) (A Record)

FIGURE 5 – Frontal facing ALB.

### 0.5.2 5.2 Load Balancer Interne (Backend)

Nom: internallb

Type: Application

Scheme: Internal

État: Active

Zone hébergée: Z355XD0TRQ7X7K

VPC: vpc-0c8323d079a5de71a

Availability Zones:

- us-east-1b (use1-az4)
- us-east-1a (use1-az2)

ARN: `arn:aws:elasticloadbalancing:us-east-1:436156172735:loadbalancer/app/internallb/b2eb5274`  
 DNS: `internal-internallb-662614552.us-east-1.elb.amazonaws.com`

#### Listener Configuration:

Protocol: HTTP

Port: 80

Action par défaut: Forward to backendservertargetgroup

Target Group Stickiness: Off

The screenshot displays the AWS Management Console interface for an internal-facing Application Load Balancer (ALB) named 'internallb'. The console shows the following details:

- Details:**
  - Load balancer type:** Application
  - Scheme:** Internal
  - Status:** Active
  - Hosted zone:** Z355XDOTRQ7X7K
  - VPC:** vpc-0c8323d079a5de71a
  - Availability Zones:** subnet-0c4b8c3b877bda410 (us-east-1b (use1-az4)), subnet-0c3fae1a8dcf764a6 (us-east-1a (use1-az2))
  - Load balancer IP address type:** IPv4
  - Date created:** December 22, 2024, 11:50 (UTC+01:00)
  - Load balancer ARN:** arn:aws:elasticloadbalancing:us-east-1:436156172735:loadbalancer/app/internallb/82eb52742dcd9c48
  - DNS name:** internal-internallb-662614552.us-east-1.elb.amazonaws.com (A Record)
- Listeners and rules:**
  - Listeners and rules (1):** A listener for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.
  - Filter listeners:** Search bar for filtering listeners.
  - Table:**

Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate
HTTP:80	Forward to target group <ul style="list-style-type: none"> <li>backendservertargetgroup (1 (100%))</li> <li>Target group stickiness: Off</li> </ul>	1 rule	ARN	Not applicable	Not applicable

FIGURE 6 – Internal facing ALB.

## 0.6 Étape 6 : Configuration de l'Auto Scaling

### 0.6.1 6.1 Auto Scaling Group Frontend

Nom: frontendautoscale

ARN: `arn:aws:autoscaling:us-east-1:436156172735:autoScalingGroup:58839559-57da-4d23-afdf-a6de`

Date de création: December 24 2024 15:16:23 GMT+0100

#### Configuration:

Capacité désirée: 2

Limites de scaling: Min 2 - Max 2

Type de capacité: Units (nombre d'instances)

#### Launch Template:

ID: lt-0b07c9e449a8455fc

Nom: frontendservertemplate

AMI: ami-0485d42d1d0d7d2cd

Type d'instance: t2.micro

Security Group: sg-09e7a872e577b8c89

## 0.6.2 6.2 Auto Scaling Group Backend

Nom: back\_end\_autoscale

ARN: arn:aws:autoscaling:us-east-1:436156172735:autoScalingGroup:6a2c1357-1d73-4680-a706-f1a1

Date de création: December 24 2024 12:52:35 GMT+0100

### Configuration:

Capacité désirée: 2

Limites de scaling: Min 2 - Max 2

Type de capacité: Units (nombre d'instances)

### Launch Template:

ID: lt-01e85cbb3421d37d1

Nom: backend\_launch\_template

AMI: ami-0f94a7308bb39e346

Type d'instance: t2.micro

Security Group: sg-03019fea108ab8e7d

## 0.7 Étape 7 : Configuration des Target Groups

### 0.7.1 7.1 Frontend Target Group

Nom: frontendservertargetgroup

Type: Instance

Protocol: HTTP

Port: 80

Version: HTTP1

### État des Cibles:

Total: 2

Healthy: 2

Unhealthy: 0

Zone: us-east-1a, us-east-1b

État: Healthy

### Load Balancer Association:

Nom: publiclb

Type: Application

Scheme: Internet-facing

Listener:

- Protocol: HTTP-80

- Action: Forward to frontendservertargetgroup (100%)

- Target group stickiness: Off

### 0.7.2 7.2 Backend Target Group

Type: Instance

Protocol: HTTP

Port: 4000

Version: HTTP1

### État des Cibles:

```

Total: 2
Healthy: 2
Unhealthy: 0
Zone: us-east-1a, us-east-1b
État: Healthy

```

#### Load Balancer Association:

```

Nom: internallb
Type: Application
Scheme: Internal
Listener:
- Protocol: HTTP-80
- Action: Forward to backendservertargetgroup (100%)
- Target group stickiness: Off

```

#### backendservertargetgroup

[Actions ▼](#)

**Details**  
[arn:aws:elasticloadbalancing:us-east-1:436156172735:targetgroup/backendservertargetgroup/ad3563e048e88cc8](#)

<b>Target type</b> Instance	<b>Protocol : Port</b> HTTP: 4000	<b>Protocol version</b> HTTP1	<b>VPC</b> <a href="#">vpc-0c8323d079a5de71a</a>
<b>IP address type</b> IPv4	<b>Load balancer</b> <a href="#">internallb</a>		

2	2	0	0	0	0
Total targets	Healthy	Unhealthy	Unused	Initial	Draining
	0 Anomalous				

► **Distribution of targets by Availability Zone (AZ)**  
Select values in this table to see corresponding filters applied to the Registered targets table below.

FIGURE 7 – Backend server target group.

#### Architecture Finale

- Infrastructure complète déployée sur plusieurs zones de disponibilité
- Séparation claire des couches applicatives (frontend, backend, base de données)
- Haute disponibilité assurée par l'Auto Scaling et la réplication Multi-AZ
- Sécurité renforcée par les groupes de sécurité et l'architecture en couches
- Load balancing efficace avec surveillance de la santé des instances
- Configuration automatisée des instances via des templates de lancement

## 0.8 Conclusion

Ce projet a permis de mettre en place une architecture cloud robuste et évolutive sur AWS. L'utilisation des différents services AWS (VPC, EC2, RDS, ELB, Auto Scaling) a permis de créer une infrastructure hautement disponible et sécurisée. La séparation en différentes couches et l'utilisation de multiples zones de disponibilité garantissent la résilience de l'application. Les bonnes pratiques de sécurité ont été suivies à travers l'utilisation de sous-réseaux privés, de groupes de sécurité et d'une instance bastion.

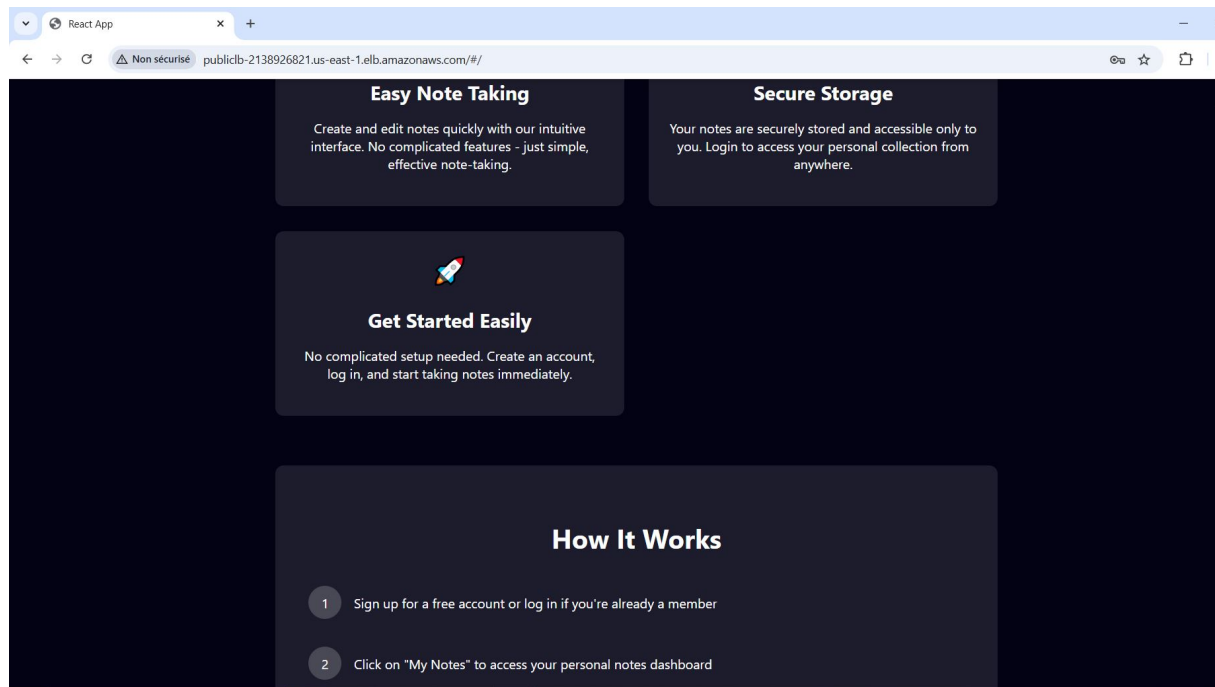


FIGURE 8 – Web App launched.

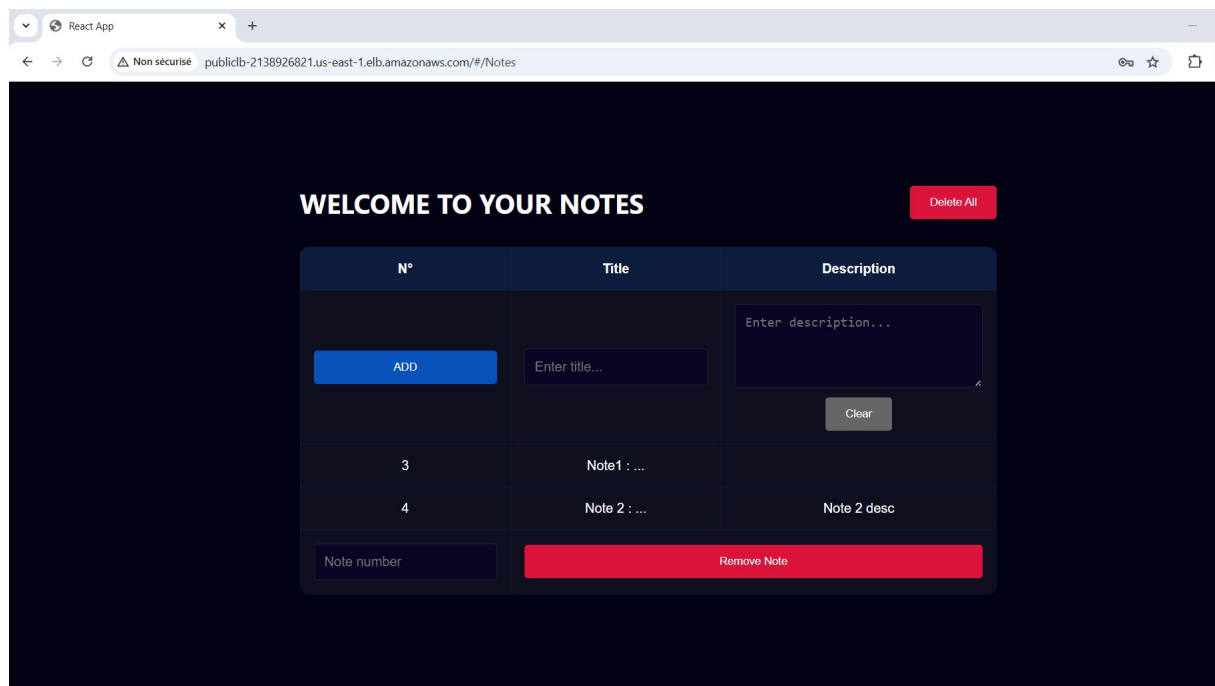


FIGURE 9 – Note Taking.