Projet

Mise en place d'un server web Flask (Python) sur 2 instances EC2, faisant appels à une base de données PostgreSQL, se trouvant dans le subnet privé du VPC, protégées par un loadbalancer se trouvant dans le subnet public du VPC.

Projet par `Vincent LAGOGUE`, `Tom THIOULOUSE`, `Alexis PLESSIAS`, `David TEJEDA` et `Thomas PEUGNET`.

Table des matières

Préparation

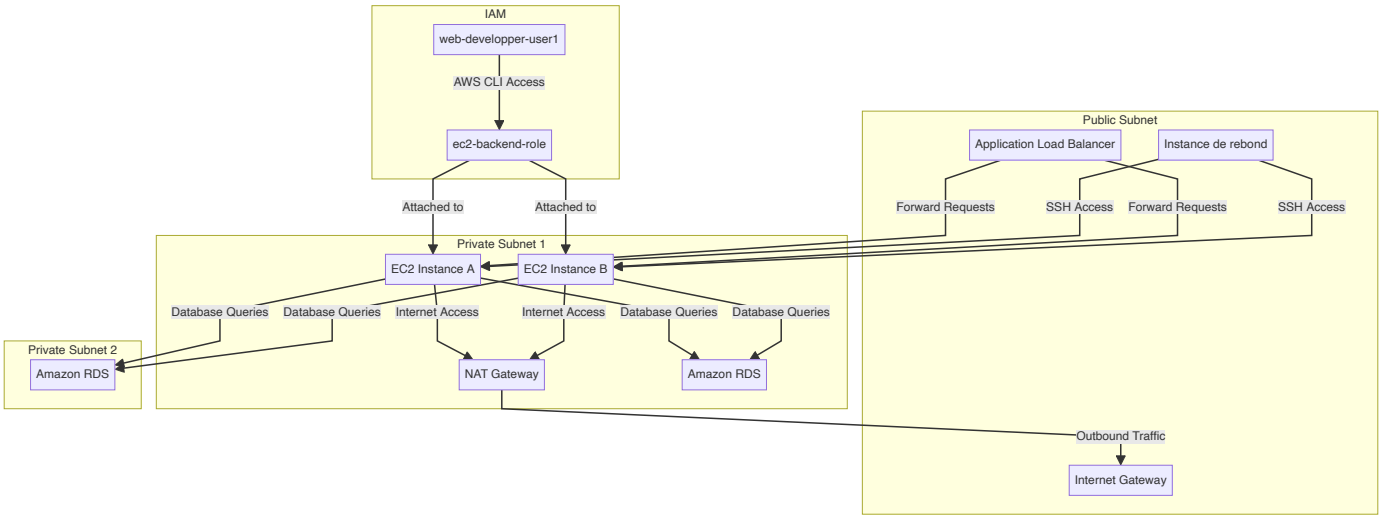
Politique SLA

- Objectif de disponibilité : 99,98 %.
- Cela implique une architecture redondante avec :
  - Un Load Balancer pour distribuer les requêtes.
  - Deux zones de disponibilité (AZ) pour les instances EC2.
  - Un système de basculement automatique en cas de panne.

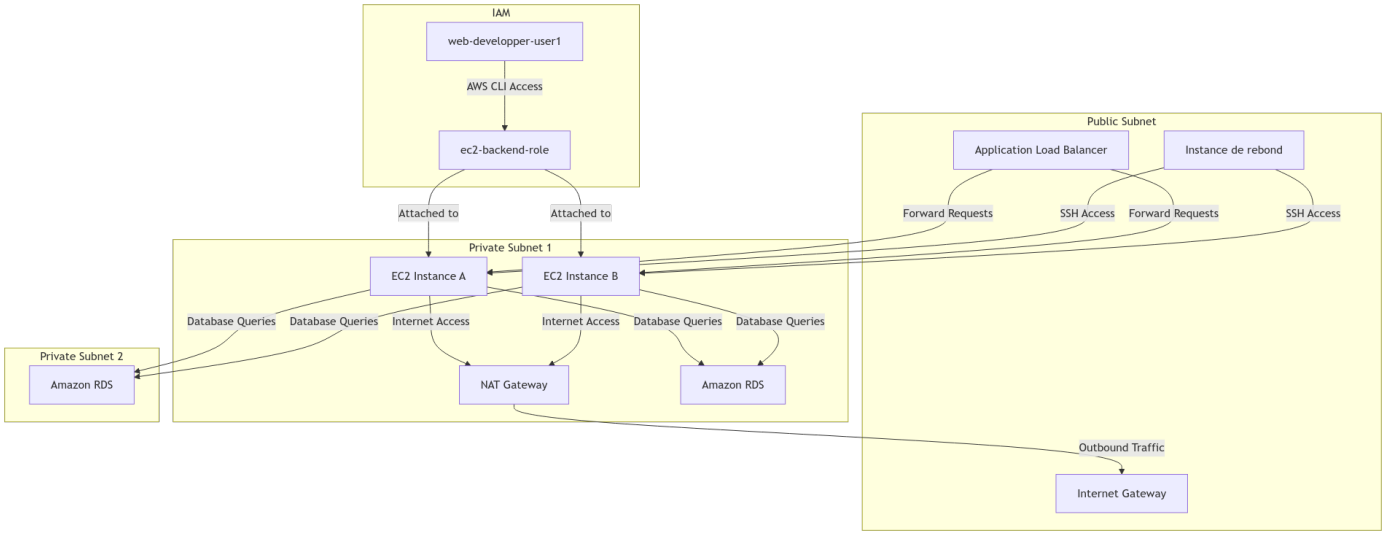
Objectifs SLA, RPO et RTO

- SLA (Disponibilité) : 99,98 %.
- RPO (Recovery Point Objective) : 5 minutes (données perdues max).
- RTO (Recovery Time Objective) : 30 minutes (temps pour restaurer les services).

Diagramme d'Architecture



Affichage en PNG ci-dessous si le code Mermaid n'est pas interprété.



Politiques de sécurité

- IAM :
  - Rôles et politiques restreints pour les instances EC2 (accès S3 minimal).
  - Rôle spécifique pour les instances EC2 (`ec2-backend-role`) permettant la gestion via AWS Systems Manager.
  - Accès sécurisé à RDS pour les administrateurs uniquement, avec des permissions limitées via des Security Groups.
- Security Groups :
  - ALB : Autorise uniquement les connexions entrantes sur HTTP (port 80) depuis `0.0.0.0/0`.
  - EC2 : Accepte les connexions uniquement depuis le ALB via son Security Group (`nlb-sg`).
  - RDS :
    - Accepte les connexions uniquement depuis les instances EC2 via leur Security Group (`ec2-sg`).
    - Appliqué à toutes les sous-routes privées hébergeant RDS (Private Subnet 1 et Private Subnet 2).
  - Instance de rebond (Rebond) :
    - Accepte les connexions entrantes uniquement sur SSH (port 22).
    - Accès SSH aux instances EC2 privées (`app-instance-1` et `app-instance-2`) via leurs Security Groups.

Surveillance et gestion des incidents

- AWS CloudWatch : Collecte des métriques système et configuration d'alertes pour détecter les indisponibilités.
- AWS CloudTrail : Suivi des journaux d’activité pour détecter des anomalies.

Tests de vulnérabilité et validation

- AWS Inspector : Analyse des failles de sécurité sur les instances EC2.
- Validation des objectifs SLA, RPO et RTO :
  - Simulation de panne sur une zone de disponibilité pour vérifier le basculement automatique.
  - Test de restauration des données depuis les snapshots RDS pour valider le RPO.

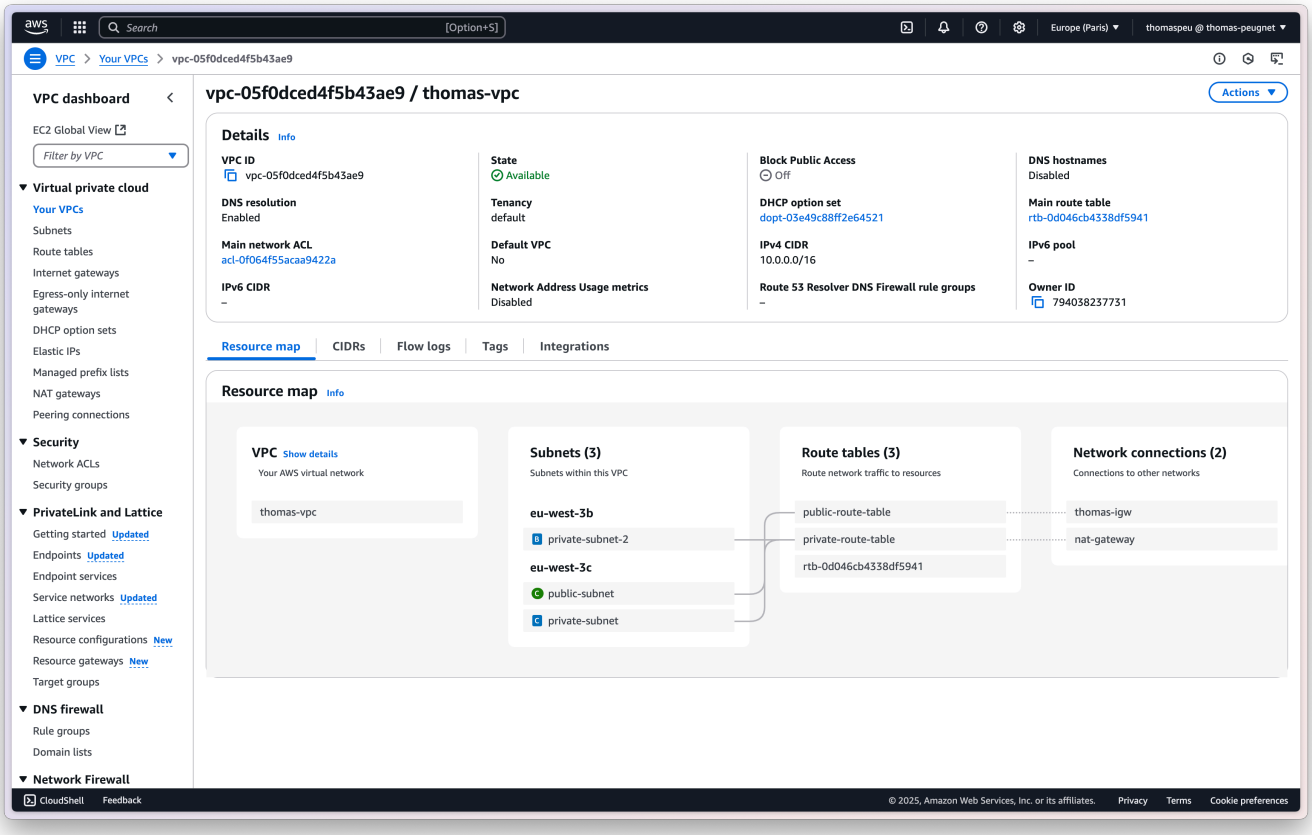
Réalisation

VPC

Nous configurons un environnement réseau dans le VPC `thomas-vpc` avec les éléments suivants :

- Subnets : `public-subnet` pour les ressources accessibles depuis Internet et `private-subnet` pour les ressources sécurisées.
- Route Tables :

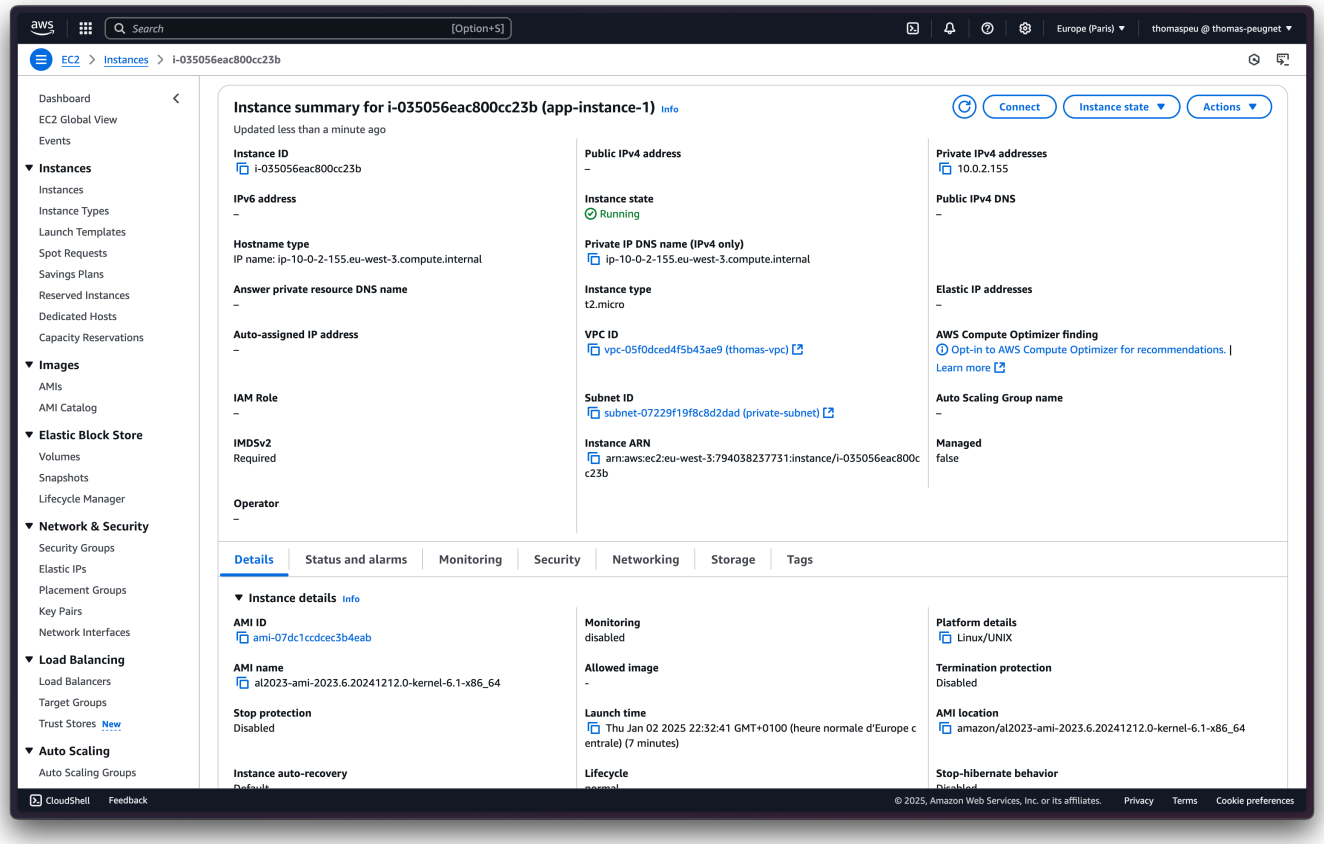
    - `public-route-table`, associée à `thomas-igw` (Internet Gateway), permet un accès Internet au sous-réseau public.
    - `private-route-table`, associée à `nat-gateway`, garantit un accès sortant sécurisé au sous-réseau privé.
- Passerelles :
    - `thomas-igw` pour l'accès Internet depuis le public-subnet.
    - `nat-gateway` pour l'accès sécurisé depuis le private-subnet.

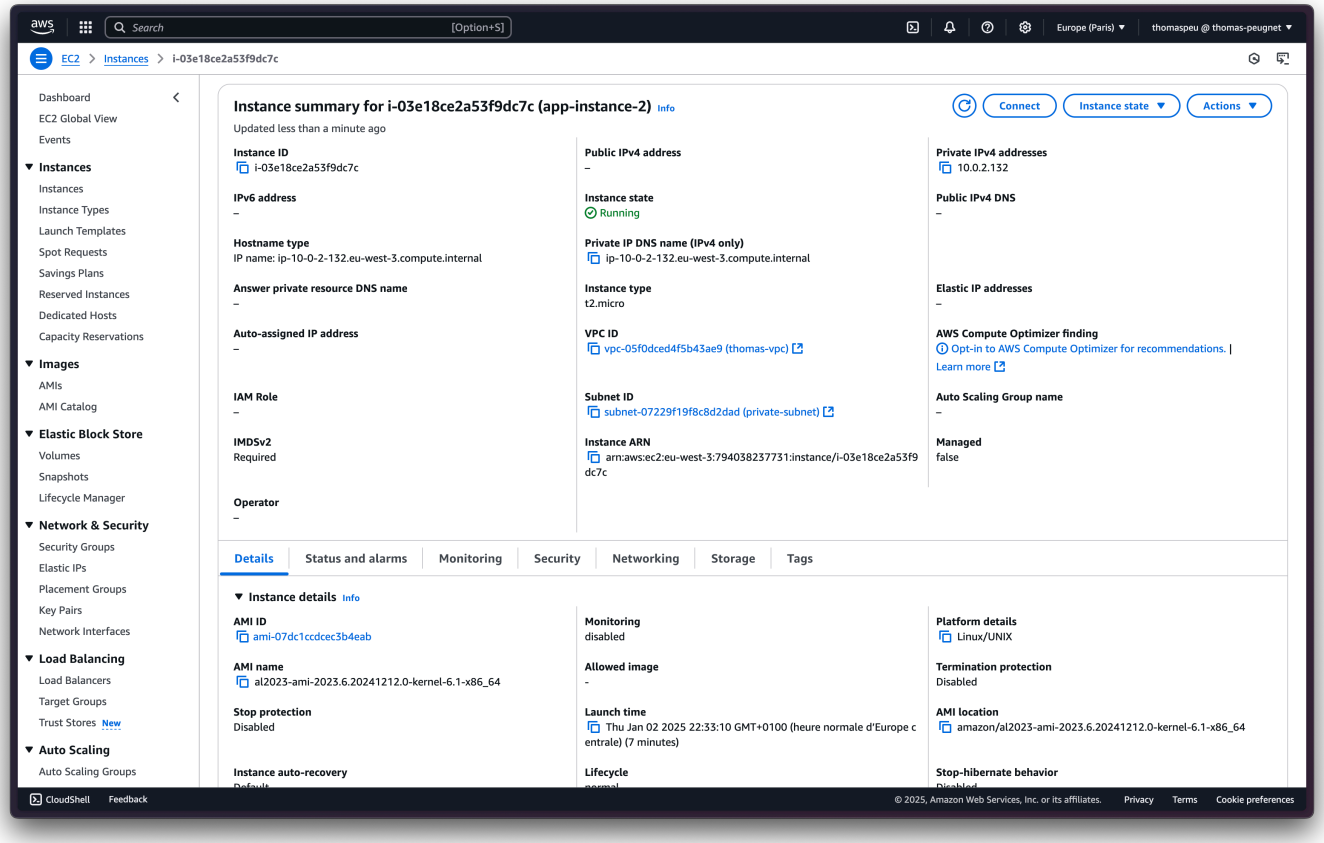


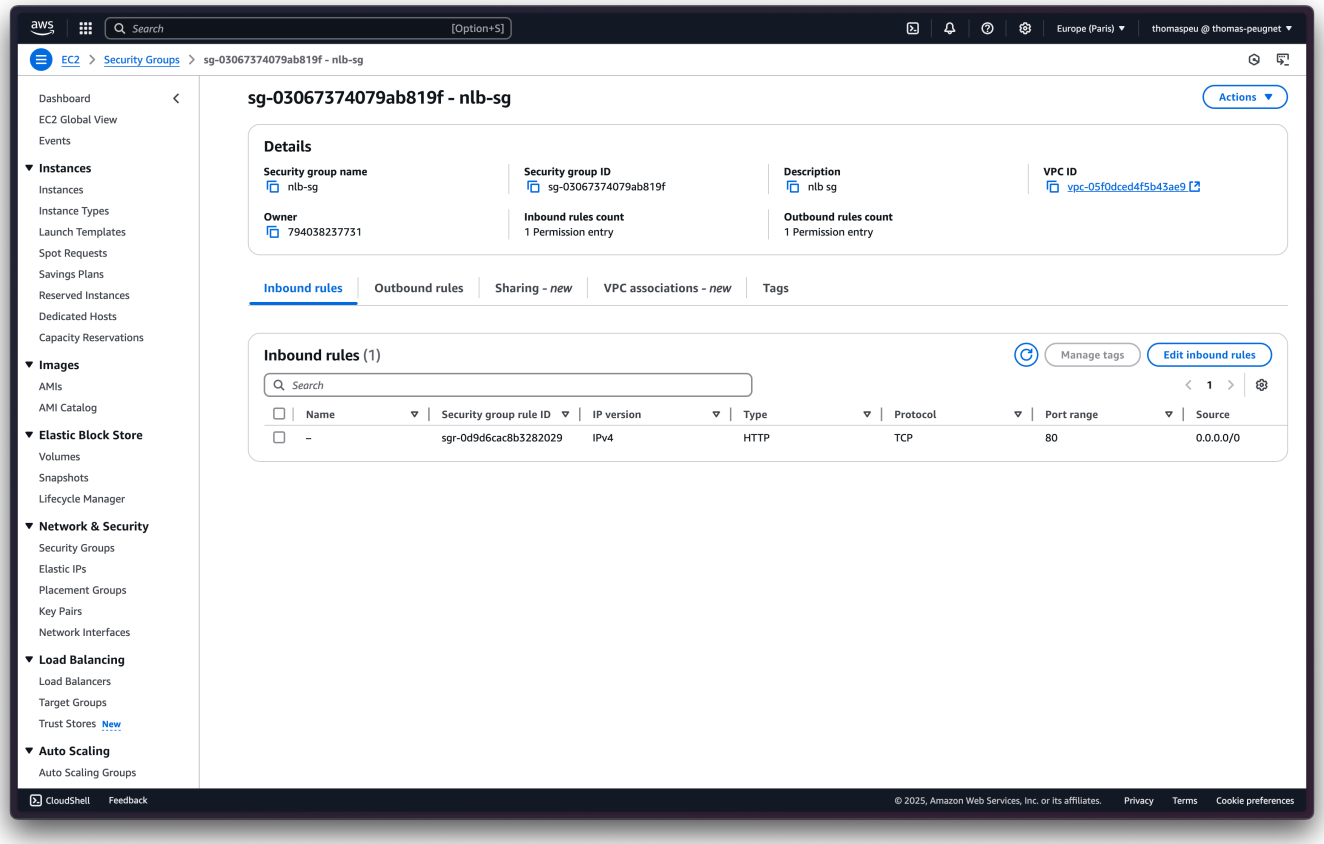
EC2 & Loadbalancer

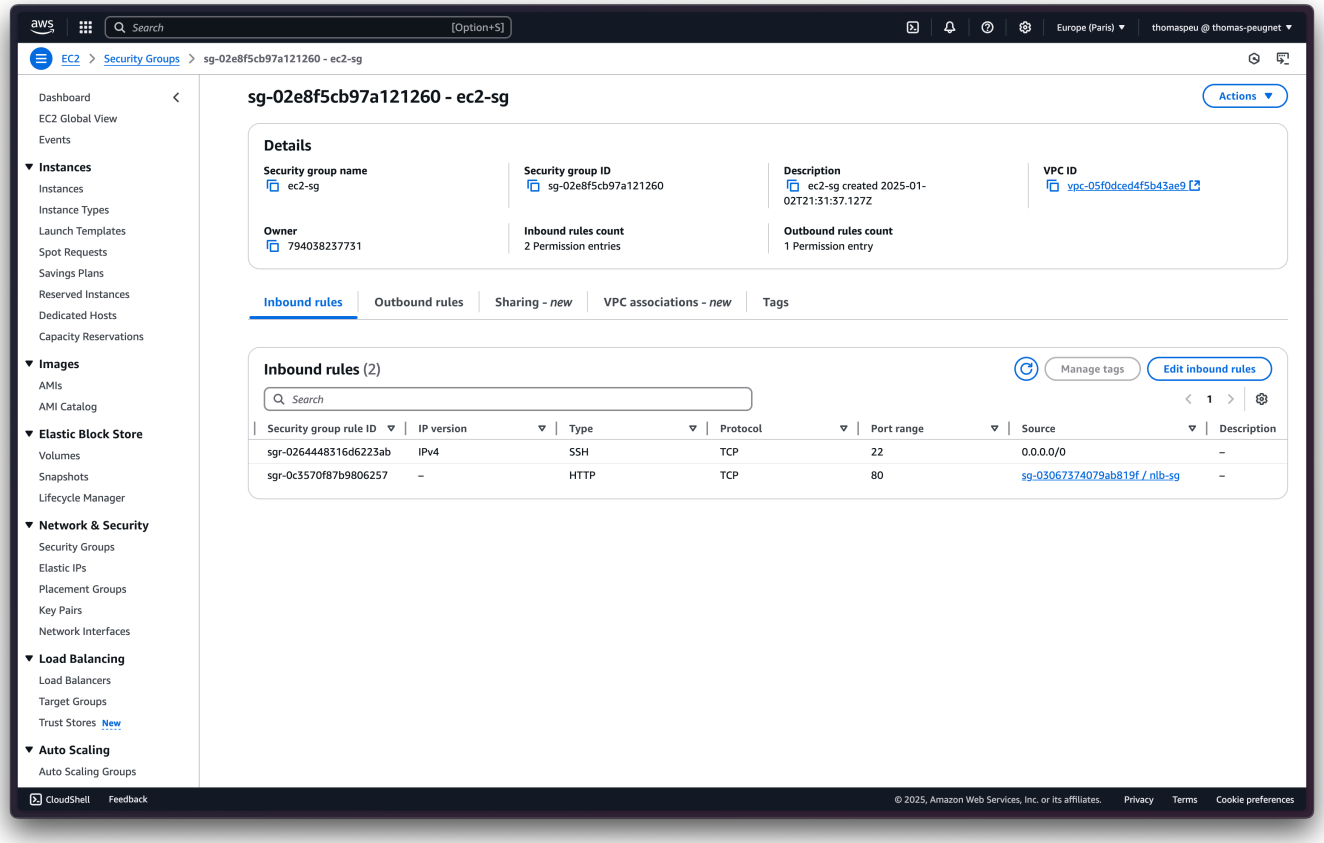
Nous configurons deux instances EC2 (`app-instance-1` et `app-instance-2`) dans le private-subnet du VPC `thomas-vpc`, accessibles via un Network Load Balancer (`app-network-load-balancer`) situé dans le public-subnet.

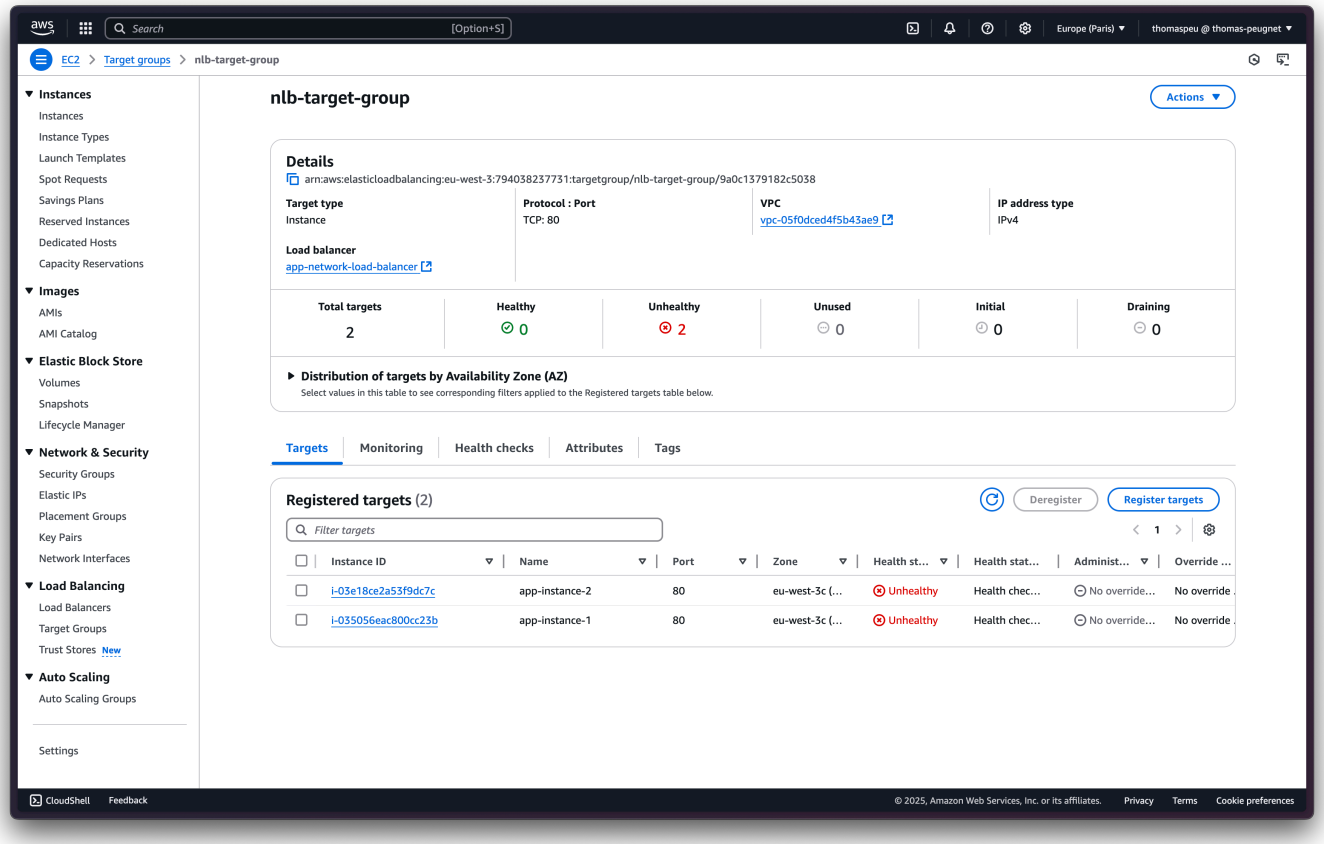
- Le NLB utilise un Target Group (`nlb-target-group`) pour distribuer le trafic TCP (port 80) vers les instances privées.
- `nlb-sg` autorise les connexions entrantes sur le NLB depuis l'extérieur, et `ec2-sg` restreint le trafic des instances à celui provenant du NLB uniquement.

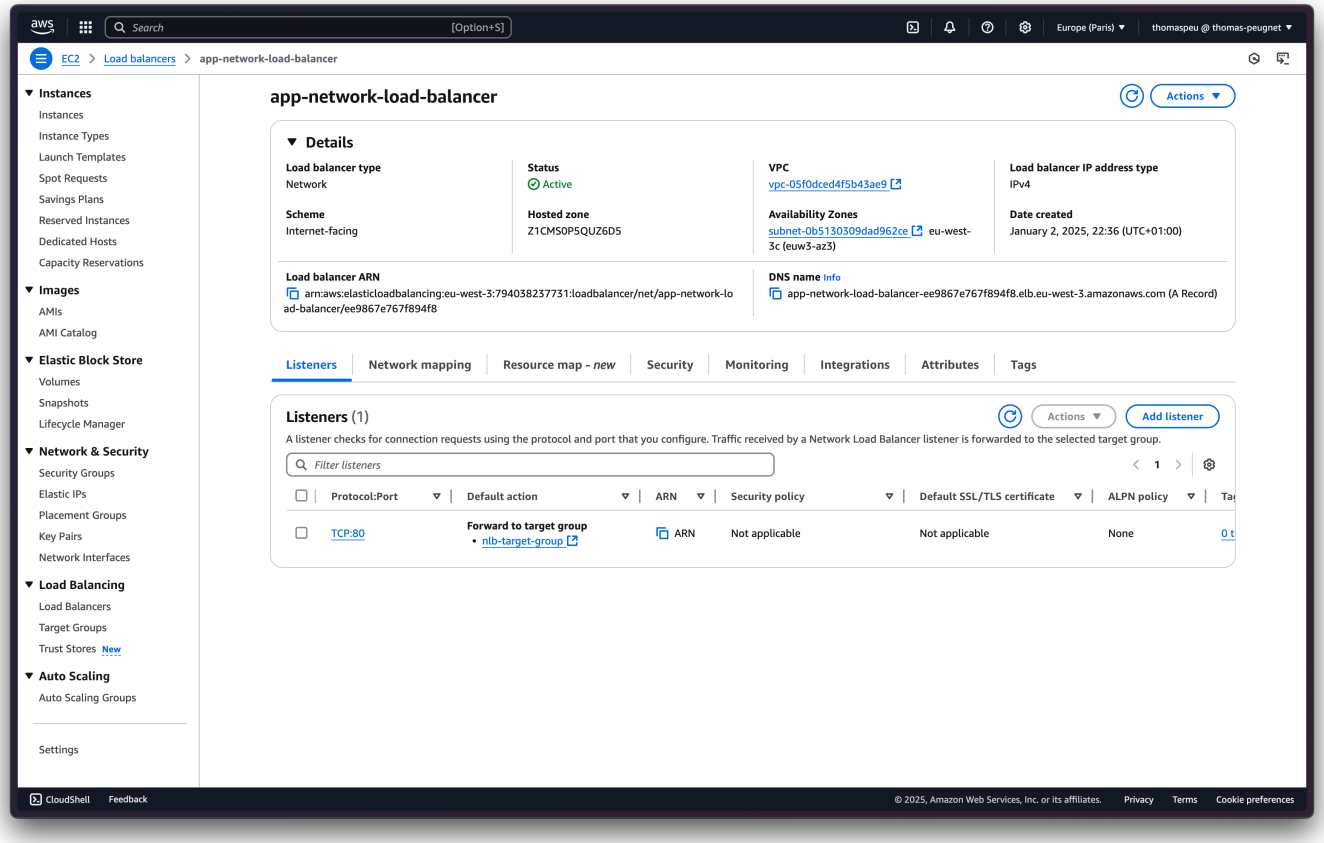

Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
EC2 > Instances > i-035056eac800cc23b
Instance summary for i-035056eac800cc23b (app-instance-1) Info
Updated less than a minute ago
Connect
Instance state
Actions
Instance ID
i-035056eac800cc23b
Public IPv4 address
–
Private IPv4 addresses
10.0.2.155
IPv6 address
–
Instance state
Running
Public IPv4 DNS
–
Hostname type
IP name: ip-10-0-2-155.eu-west-3.compute.internal
Private IP DNS name (IPv4 only)
ip-10-0-2-155.eu-west-3.compute.internal
Answer private resource DNS name
–
Instance type
t2.micro
Elastic IP addresses
–
Auto-assigned IP address
–
VPC ID
vpc-05f0dced4f5b43ae9 (thomas-vpc)
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
Learn more
IAM Role
–
Subnet ID
subnet-07229f19f8c8d2dad (private-subnet)
Auto Scaling Group name
–
IMDSv2
Required
Instance ARN
arn:aws:ec2:eu-west-3:794038237731:instance/i-035056eac800cc23b
Managed
false
Operator
–
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Instance details Info
AMI ID
ami-07dc1ccdcec3b4eab
Monitoring
disabled
Platform details
Linux/UNIX
AMI name
al2023-ami-2023.6.20241212.0-kernel-6.1-x86_64
Allowed image
-
Termination protection
Disabled
Stop protection
Disabled
Launch time
Thu Jan 02 2025 22:32:41 GMT+0100 (heure normale d'Europe centrale) (7 minutes)
AMI location
amazon/al2023-ami-2023.6.20241212.0-kernel-6.1-x86_64
Instance auto-recovery
Lifecycle
Stop-hibernate behavior
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
EC2 > Instances > i-03e18ce2a53f9dc7c
Instance summary for i-03e18ce2a53f9dc7c (app-instance-2) Info
Updated less than a minute ago
Connect
Instance state
Actions
Instance ID
i-03e18ce2a53f9dc7c
Public IPv4 address
–
Private IPv4 addresses
10.0.2.132
IPv6 address
–
Instance state
Running
Public IPv4 DNS
–
Hostname type
IP name: ip-10-0-2-132.eu-west-3.compute.internal
Private IP DNS name (IPv4 only)
ip-10-0-2-132.eu-west-3.compute.internal
Answer private resource DNS name
–
Instance type
t2.micro
Elastic IP addresses
–
Auto-assigned IP address
–
VPC ID
vpc-05f0dced4f5b43ae9 (thomas-vpc)
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
Learn more
IAM Role
–
Subnet ID
subnet-07229f19f8c8d2dad (private-subnet)
Auto Scaling Group name
–
IMDSv2
Required
Instance ARN
arn:aws:ec2:eu-west-3:794038237731:instance/i-03e18ce2a53f9dc7c
Managed
false
Operator
–
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Instance details Info
AMI ID
ami-07dc1ccdcec3b4eab
Monitoring
disabled
Platform details
Linux/UNIX
AMI name
al2023-ami-2023.6.20241212.0-kernel-6.1-x86_64
Allowed image
-
Termination protection
Disabled
Stop protection
Disabled
Launch time
Thu Jan 02 2025 22:33:10 GMT+0100 (heure normale d'Europe centrale) (7 minutes)
AMI location
amazon/al2023-ami-2023.6.20241212.0-kernel-6.1-x86_64
Instance auto-recovery
Lifecycle
Stop-hibernate behavior
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
EC2 > Security Groups > sg-03067374079ab819f - nlb-sg
sg-03067374079ab819f - nlb-sg
Actions
Details
Security group name
nlb-sg
Security group ID
sg-03067374079ab819f
Description
nlb sg
VPC ID
vpc-05f0dced4f5b43ae9
Owner
794038237731
Inbound rules count
1 Permission entry
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing - new
VPC associations - new
Tags
Inbound rules (1)
Manage tags
Edit inbound rules
Search
Name
Security group rule ID
IP version
Type
Protocol
Port range
Source
–
sgr-0d9d6cac8b3282029
IPv4
HTTP
TCP
80
0.0.0.0/0
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Target Groups
Trust Stores New
Auto Scaling
Auto Scaling Groups
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
EC2 > Security Groups > sg-02e8f5cb97a121260 - ec2-sg
sg-02e8f5cb97a121260 - ec2-sg
Actions
Details
Security group name
ec2-sg
Security group ID
sg-02e8f5cb97a121260
Description
ec2-sg created 2025-01-02T21:31:37.127Z
VPC ID
vpc-05f0dced4f5b43ae9
Owner
794038237731
Inbound rules count
2 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing - new
VPC associations - new
Tags
Inbound rules (2)
Manage tags
Edit inbound rules
Search
Security group rule ID
IP version
Type
Protocol
Port range
Source
Description
sgr-0264448316d6223ab
IPv4
SSH
TCP
22
0.0.0.0/0
–
sgr-0c3570f87b9806257
–
HTTP
TCP
80
sg-03067374079ab819f / nlb-sg
–
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Target Groups
Trust Stores New
Auto Scaling
Auto Scaling Groups
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

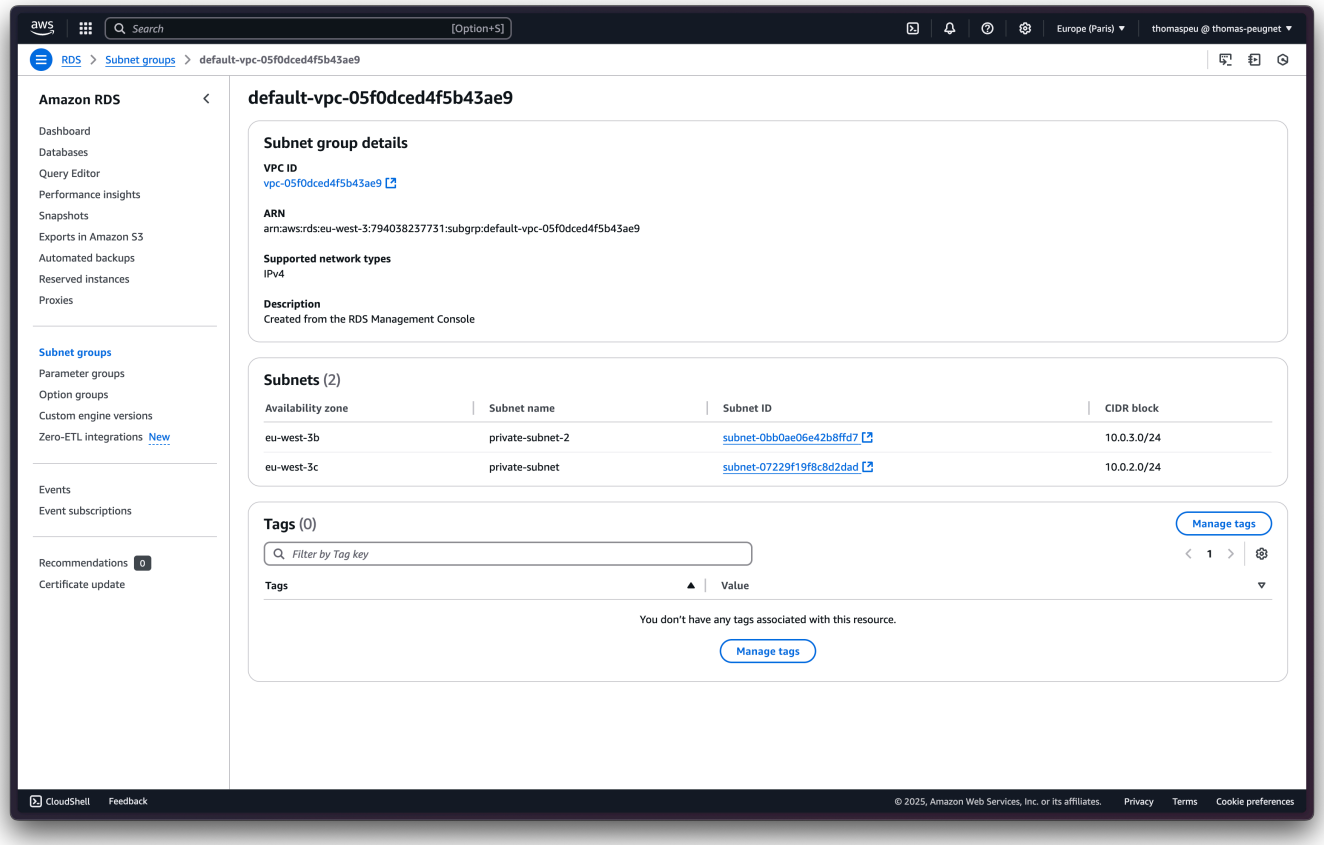

Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
EC2 > Target groups > nlb-target-group
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Target Groups
Trust Stores New
Auto Scaling
Auto Scaling Groups
Settings
nlb-target-group
Actions
Details
arn:aws:elasticloadbalancing:eu-west-3:794038237731:targetgroup/nlb-target-group/9a0c1379182c5038
Target type
Instance
Protocol : Port
TCP: 80
VPC
vpc-05f0dced4f5b43ae9
IP address type
IPv4
Load balancer
app-network-load-balancer
Total targets
2
Healthy
0
Unhealthy
2
Unused
0
Initial
0
Draining
0
Distribution of targets by Availability Zone (AZ)
Select values in this table to see corresponding filters applied to the Registered targets table below.
Targets
Monitoring
Health checks
Attributes
Tags
Registered targets (2)
Deregister
Register targets
Filter targets
Instance ID
Name
Port
Zone
Health st...
Health stat...
Administ...
Override ...
i-03e18ce2a53f9dc7c
app-instance-2
80
eu-west-3c (...
Unhealthy
Health chec...
No override...
No override
i-035056eac800cc23b
app-instance-1
80
eu-west-3c (...
Unhealthy
Health chec...
No override...
No override
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

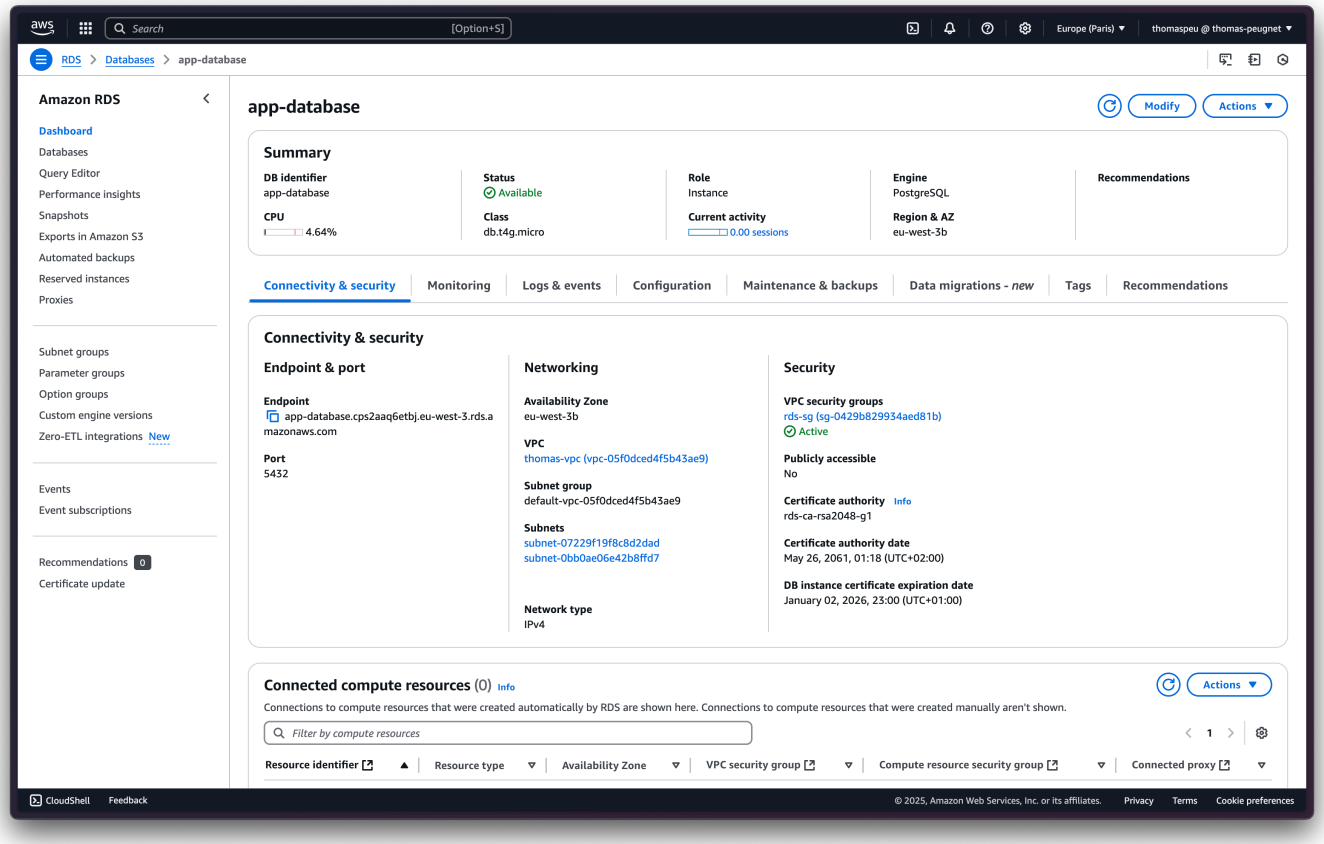

Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
EC2 > Load balancers > app-network-load-balancer
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Target Groups
Trust Stores New
Auto Scaling
Auto Scaling Groups
Settings
app-network-load-balancer
Actions
Details
Load balancer type
Network
Status
Active
VPC
vpc-05f0dced4f5b43ae9
Load balancer IP address type
IPv4
Scheme
Internet-facing
Hosted zone
Z1CMS0P5QUZ6D5
Availability Zones
subnet-0b5130309dad962ce eu-west-3c (euw3-az3)
Date created
January 2, 2025, 22:36 (UTC+01:00)
Load balancer ARN
arn:aws:elasticloadbalancing:eu-west-3:794038237731:loadbalancer/net/app-network-load-balancer/ee9867e767f894f8
DNS name Info
app-network-load-balancer-ee9867e767f894f8.elb.eu-west-3.amazonaws.com (A Record)
Listeners
Network mapping
Resource map - new
Security
Monitoring
Integrations
Attributes
Tags
Listeners (1)
Actions
Add listener
A listener checks for connection requests using the protocol and port that you configure. Traffic received by a Network Load Balancer listener is forwarded to the selected target group.
Filter listeners
Protocol:Port
Default action
ARN
Security policy
Default SSL/TLS certificate
ALPN policy
TCP:80
Forward to target group
nlb-target-group
ARN
Not applicable
Not applicable
None
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

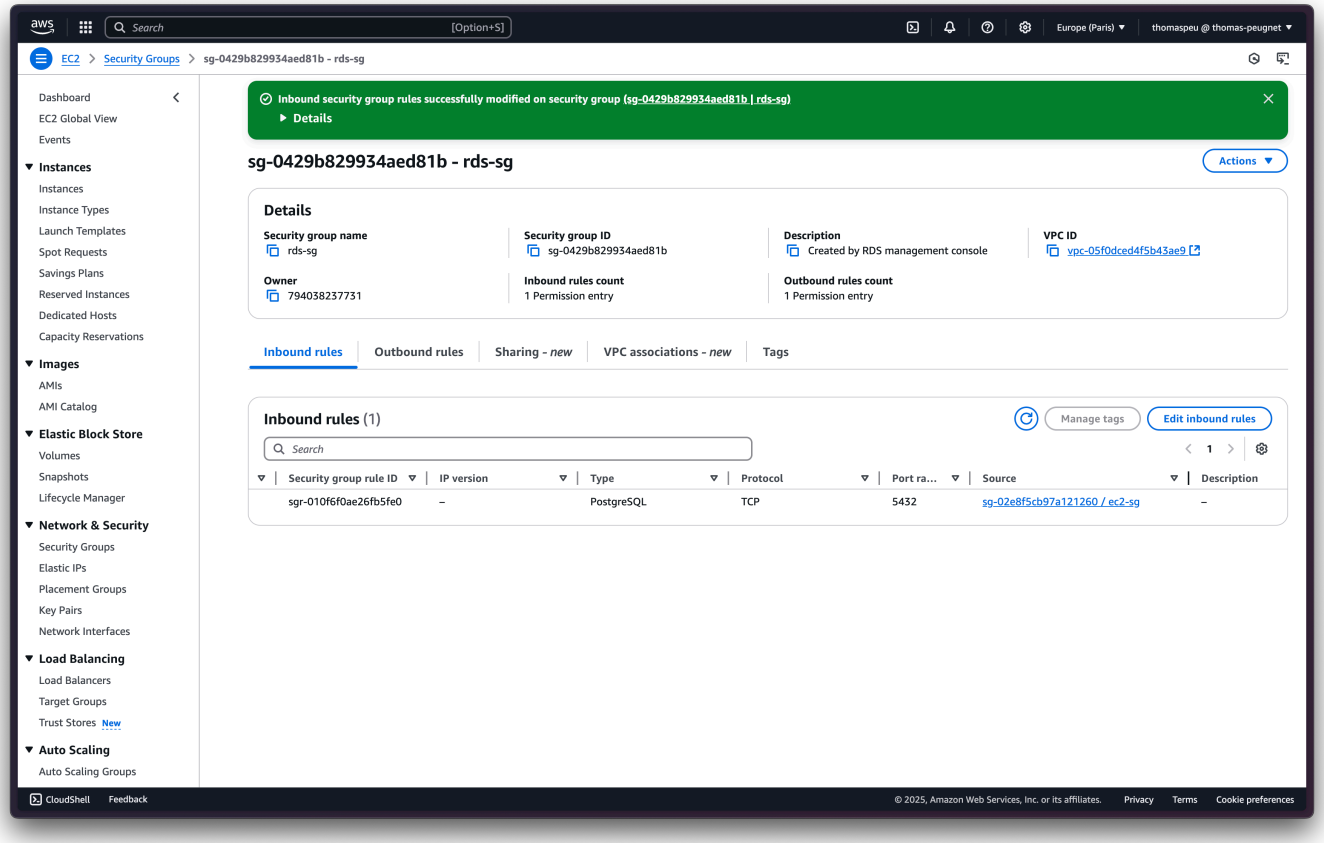
RDS

Nous configurons une base de données RDS PostgreSQL (`app-database`) dans le VPC `thomas-vpc`, utilisant un DB Subnet Group couvrant les sous-réseaux privés (`private-subnet` et `private-subnet-2`, ce dernier ayant été créé pour respecter les contraintes d'AWS, la database ne doit pas être accessible depuis seulement une seule AZ).

Le Security Group de RDS (`rds-sg`) ayant été automatiquement créé est ajusté : toute règle autorisant une IP publique est supprimée, et une règle autorisant les connexions depuis les instances EC2 via leur Security Group (`ec2-sg`) est ajoutée. La base de données est isolée et uniquement accessible depuis les ressources internes du VPC.

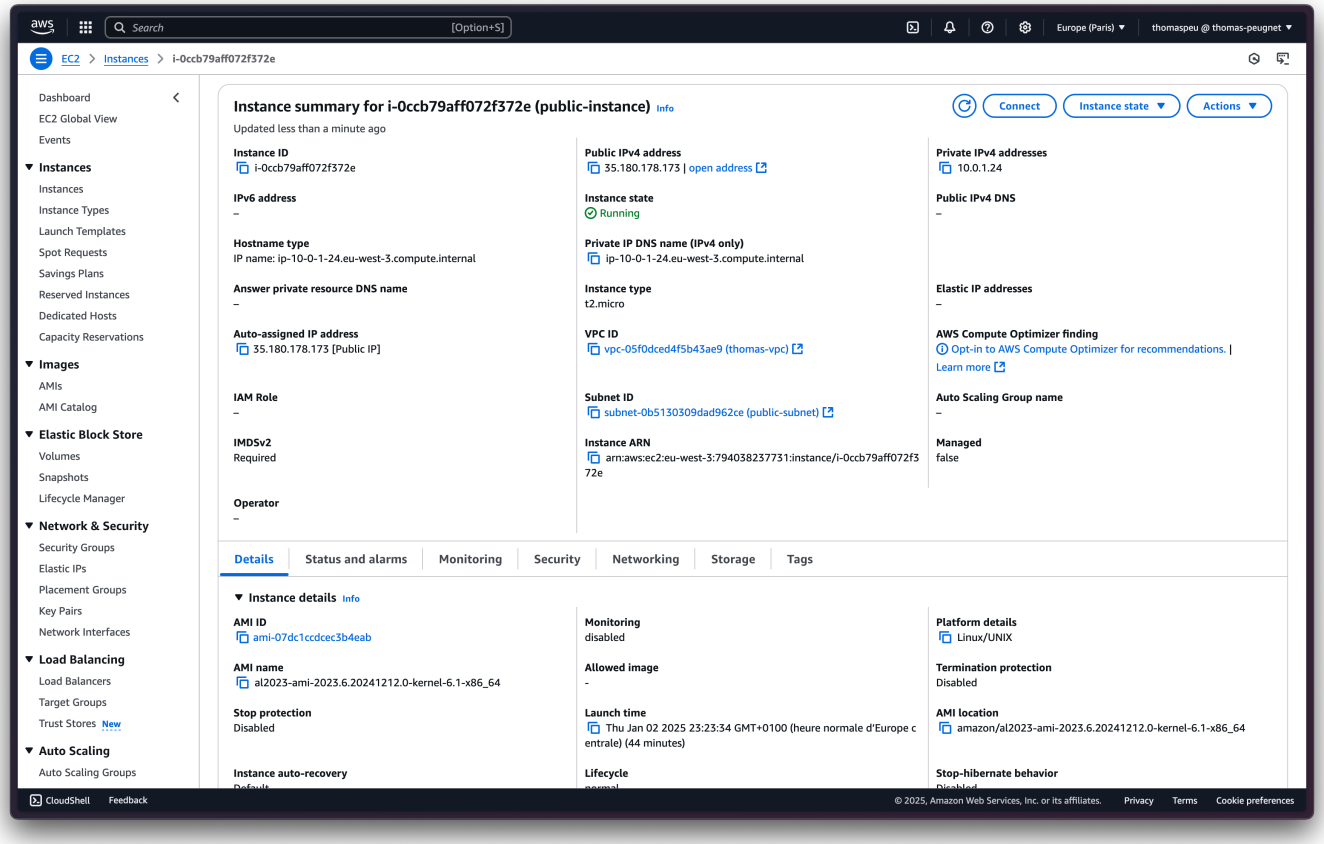



Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
RDS > Databases > app-database
Amazon RDS
Dashboard
Databases
Query Editor
Performance insights
Snapshots
Exports in Amazon S3
Automated backups
Reserved instances
Proxies
Subnet groups
Parameter groups
Option groups
Custom engine versions
Zero-ETL integrations New
Events
Event subscriptions
Recommendations 0
Certificate update
app-database
Modify
Actions
Summary
DB identifier
app-database
Status
Available
Role
Instance
Engine
PostgreSQL
Recommendations
CPU
4.64%
Class
db.t4g.micro
Current activity
0.00 sessions
Region & AZ
eu-west-3b
Connectivity & security
Monitoring
Logs & events
Configuration
Maintenance & backups
Data migrations - new
Tags
Recommendations
Connectivity & security
Endpoint & port
Endpoint
app-database.cps2aaq6etbj.eu-west-3.rds.amazonaws.com
Port
5432
Networking
Availability Zone
eu-west-3b
VPC
thomas-vpc (vpc-05f0dced4f5b43ae9)
Subnet group
default-vpc-05f0dced4f5b43ae9
Subnets
subnet-07229f19f8c8d2dad
subnet-0bb0ae06e42b8ffd7
Network type
IPv4
Security
VPC security groups
rds-sg (sg-0429b829934aed81b)
Active
Publicly accessible
No
Certificate authority Info
rds-ca-rsa2048-g1
Certificate authority date
May 26, 2061, 01:18 (UTC+02:00)
DB instance certificate expiration date
January 02, 2026, 23:00 (UTC+01:00)
Connected compute resources (0) Info
Actions
Connections to compute resources that were created automatically by RDS are shown here. Connections to compute resources that were created manually aren't shown.
Filter by compute resources
Resource identifier
Resource type
Availability Zone
VPC security group
Compute resource security group
Connected proxy
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
EC2 > Security Groups > sg-0429b829934aed81b - rds-sg
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Target Groups
Trust Stores New
Auto Scaling
Auto Scaling Groups
Inbound security group rules successfully modified on security group (sg-0429b829934aed81b | rds-sg)
Details
sg-0429b829934aed81b - rds-sg
Actions
Details
Security group name
rds-sg
Security group ID
sg-0429b829934aed81b
Description
Created by RDS management console
VPC ID
vpc-05f0dced4f5b43ae9
Owner
794038237731
Inbound rules count
1 Permission entry
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing - new
VPC associations - new
Tags
Inbound rules (1)
Manage tags
Edit inbound rules
Search
Security group rule ID
IP version
Type
Protocol
Port ra...
Source
Description
sgr-010f6f0ae26fb5fe0
–
PostgreSQL
TCP
5432
sg-02e8f5cb97a121260 / ec2-sg
–
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Configuration du backend Flask et PostgreSQL

Nous avons créé une nouvelle instance EC2 pour faire rebond et permettre l'accès aux instances dans le `private-subnet`. Cette instance est nommée `public-instance`.



Installation des dépendances sur les 2 instances EC2 dans le subnet privé :

- Mise à jour des paquets et installation de Python 3, `pip`, et des bibliothèques nécessaires (`flask`, `psycopg2-binary`).
- Installation des outils PostgreSQL (`postgresql16`) pour interagir avec la base RDS.

Configuration de l'application Flask :

- Création d'un fichier `app.py` pour implémenter un serveur web minimal et établir une connexion avec la base de données RDS PostgreSQL.
- Ajout d'une route principale (`/`) pour vérifier que l'application tourne et d'une route (`/data`) pour tester les interactions avec la base de données.

Gestion des variables d'environnement :

- Configuration d'un fichier `.env` contenant les informations sensibles comme l'endpoint RDS, le nom de la base, l'utilisateur, et le mot de passe.
- Chargement des variables d'environnement dans l'environnement d'exécution Flask.

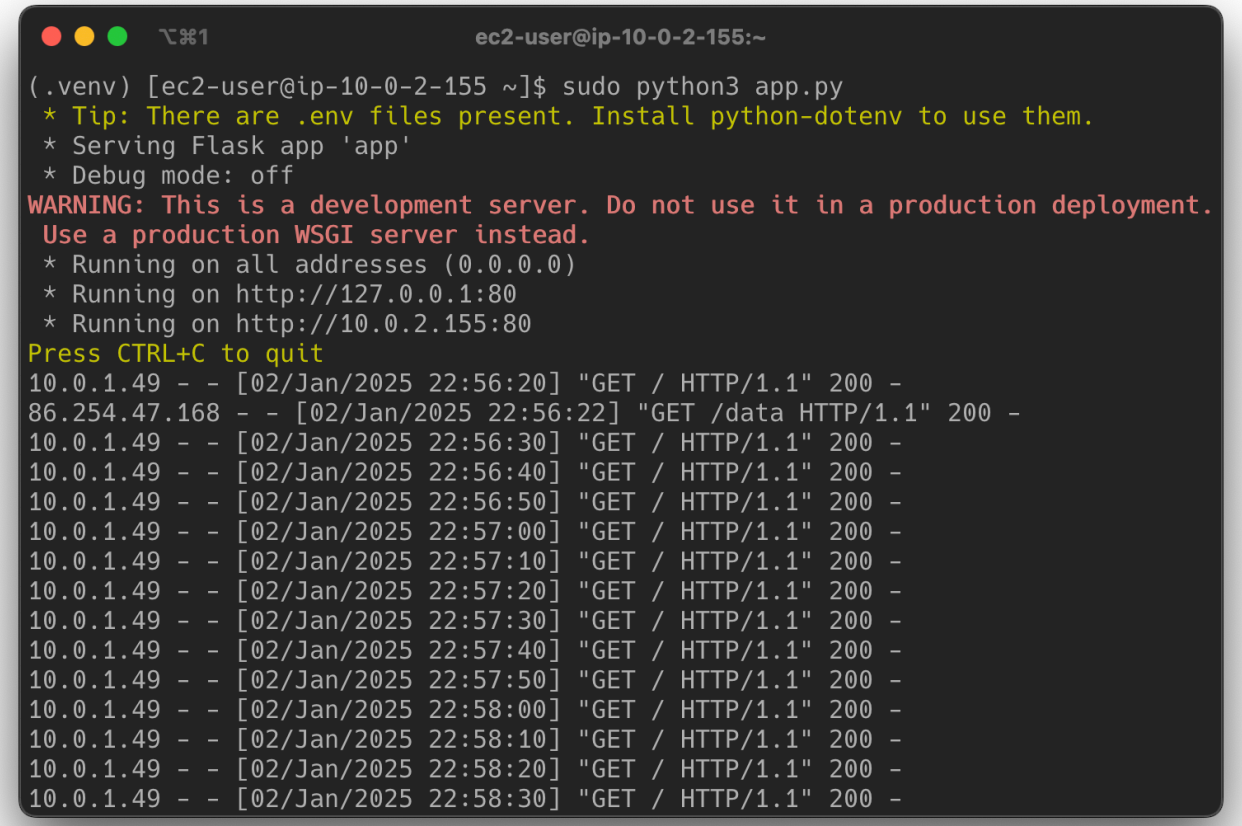
Le code Python est le suivant.

```
from flask import Flask, jsonify
```

```python
import psycopg2
import os

app = Flask(__name__)

DB_HOST = os.getenv("DB_HOST")
DB_NAME = os.getenv("DB_NAME")
DB_USER = os.getenv("DB_USER")
DB_PASSWORD = os.getenv("DB_PASSWORD")

def get_db_connection():
    try:
        conn = psycopg2.connect(
            host=DB_HOST,
            database=DB_NAME,
            user=DB_USER,
            password=DB_PASSWORD
        )
        return conn
    except Exception as e:
        print(f"Connection to the database failed : {e}")
        return None

@app.route("/")
def home():
    return "Flask App is running!"

@app.route("/data")
def fetch_data():
    conn = get_db_connection()
    if conn is None:
        return jsonify({"error": "Connection to the database failed"}), 500

    cur = conn.cursor()
    cur.execute("SELECT NOW();")
    result = cur.fetchone()
    cur.close()
    conn.close()
    return jsonify({"current_time": result[0]})

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=80)
```

Lancement de Flask :

- Exécution de l'application Flask sur le port 80 avec le paramètre `host="0.0.0.0"` pour permettre l'accès depuis d'autres machines dans le réseau.

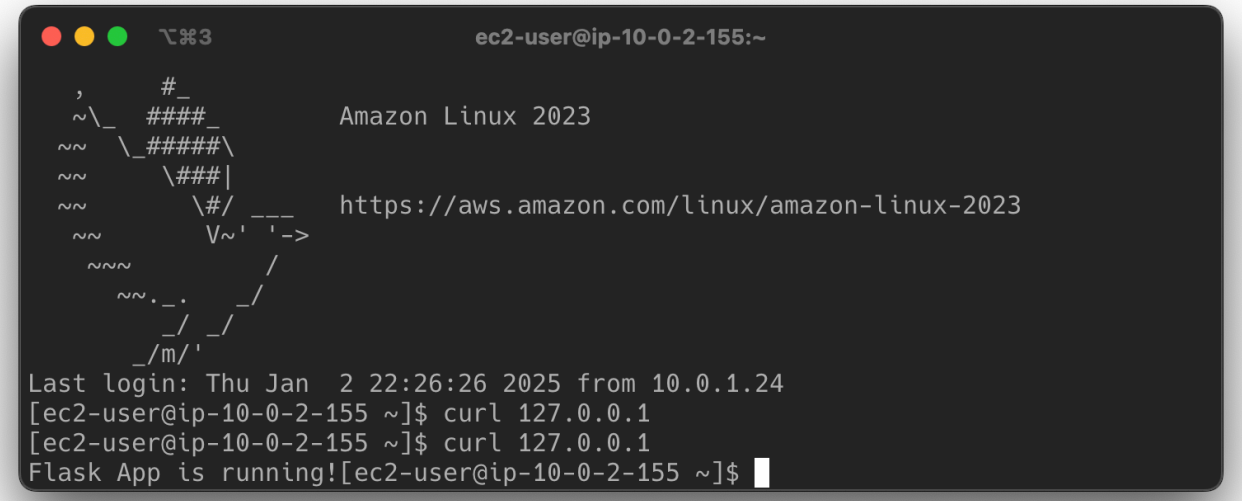


Toutes les commandes qui auront été tapées pour installer correctement PostgreSQL, les bonnes dépendances, tester l'installation et lancer l'application se trouvent ci-dessous.

```
vim app.py
sudo yum install python3-pip
sudo yum install -y python3-devel gcc postgresql-devel
sudo pip3 install psycopg2 flask
python3 -c "import psycopg2; print(psycopg2.__version__)"
sudo yum install -y postgresql16
psql -h app-database.cps2aaq6etbj.eu-west-3.rds.amazonaws.com -U postgres -d postgres
sudo python3 app.py
vim app.py
sudo python3 app.py
```

Tests locaux sur les instances :

- Vérification que l'application répond correctement aux requêtes en accédant directement depuis l'instance elle-même (`127.0.0.1:80`).



Configuration de la connectivité publique :

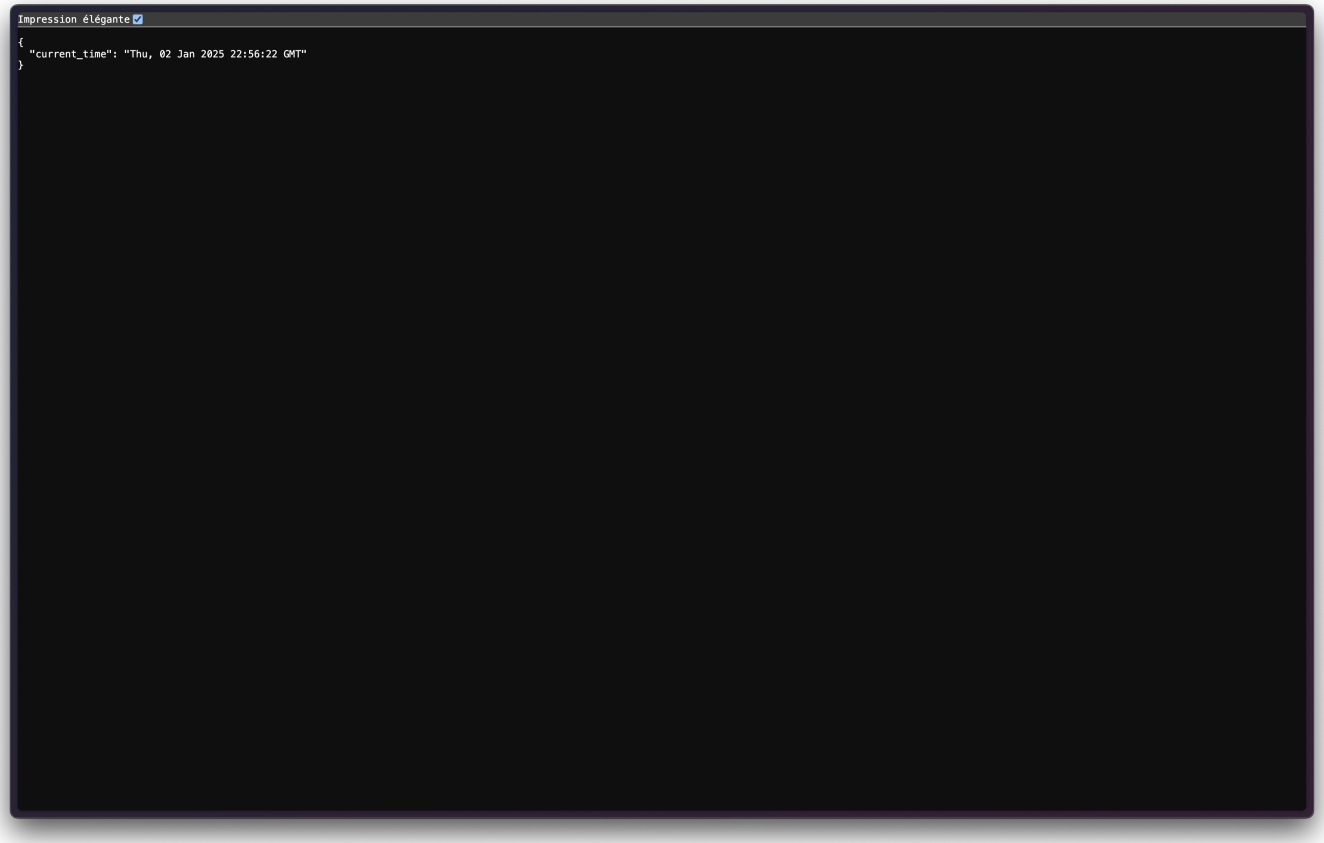
- Intégration des instances EC2 dans un Target Group associé au Network Load Balancer.
- Vérification des règles des Security Groups :
  - Le Load Balancer autorise le trafic HTTP (port 80) depuis `0.0.0.0/0`.
  - Les instances EC2 autorisent uniquement le trafic HTTP provenant du Security Group du Load Balancer.

Test final via le Load Balancer :

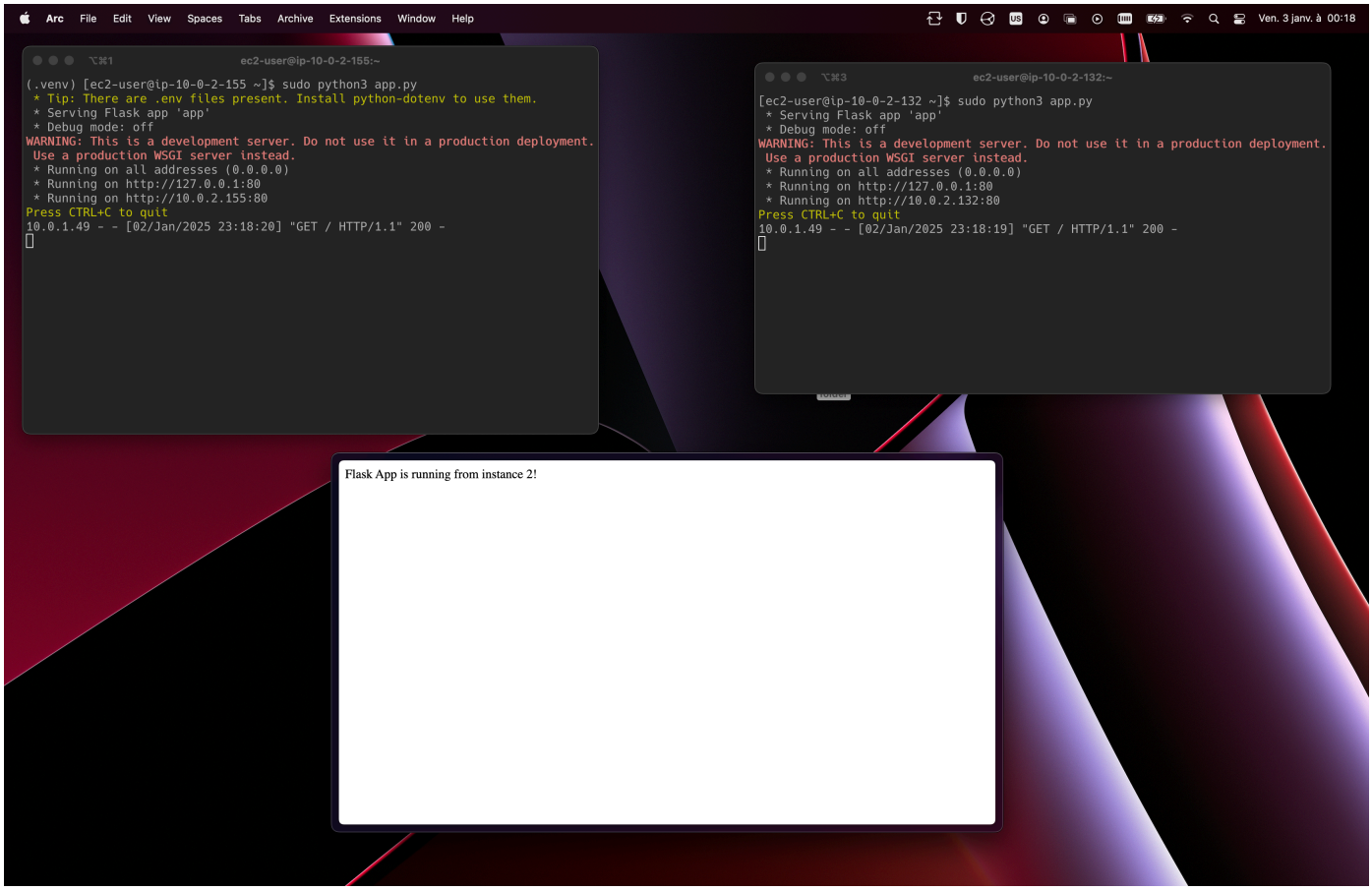
- Accès public à l'application via l'adresse DNS du Load Balancer.
- Validation des routes Flask (`/` et `/data`) pour confirmer que l'application et la base de données fonctionnent correctement.

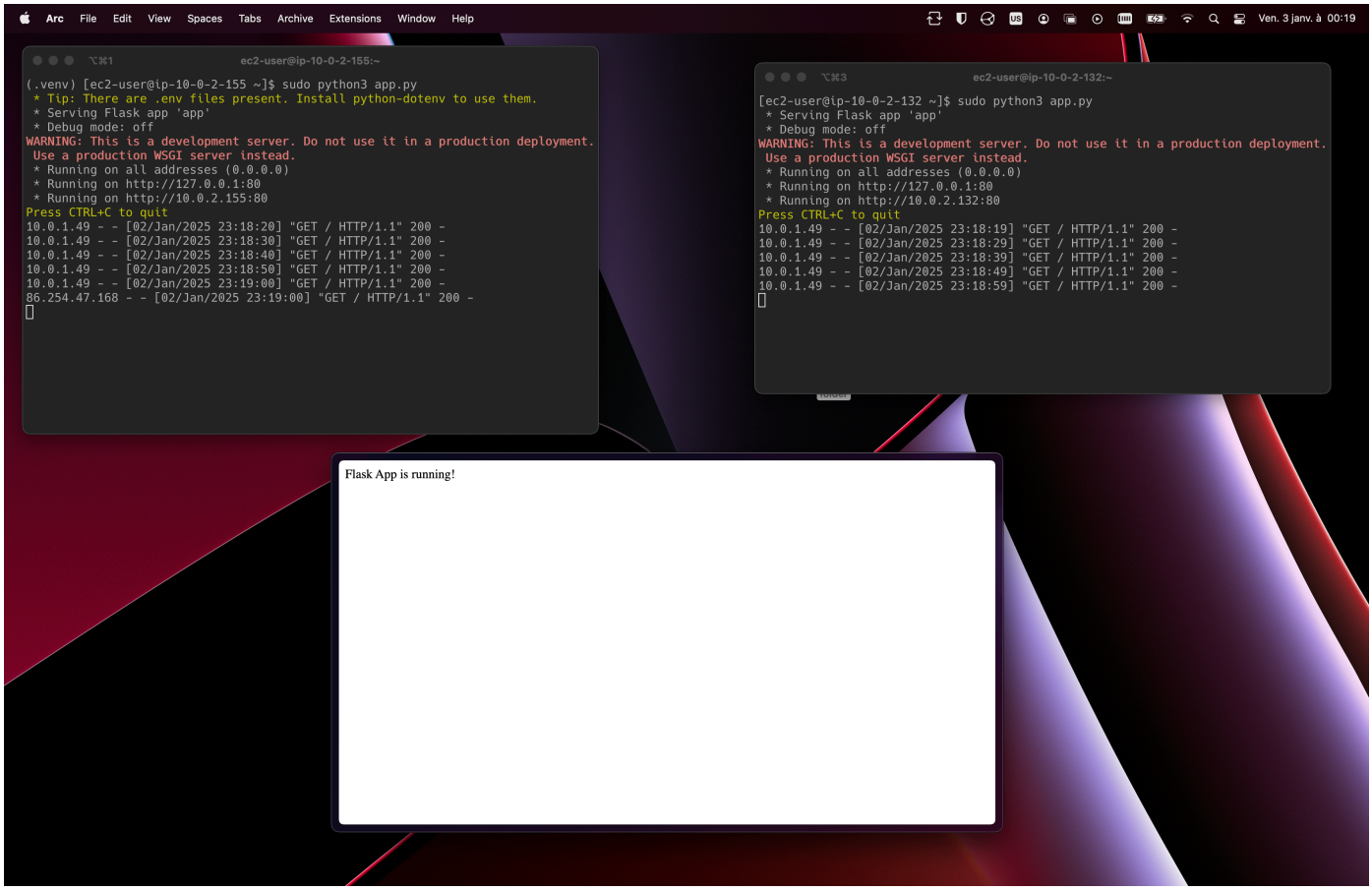
Sur `http://app-network-load-balancer-ee9867e767f894f8.elb.eu-west-3.amazonaws.com/` :

Sur `http://app-network-load-balancer-ee9867e767f894f8.elb.eu-west-3.amazonaws.com/data` :



En rafraîchissant plusieurs fois la page, nous constatons une alternance entre les différentes instances touchées.


Arc File Edit View Spaces Tabs Archive Extensions Window Help
Ven. 3 janv. à 00:18
ec2-user@ip-10-0-2-155:~
(.venv) [ec2-user@ip-10-0-2-155 ~]$ sudo python3 app.py
* Tip: There are .env files present. Install python-dotenv to use them.
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:80
* Running on http://10.0.2.155:80
Press CTRL+C to quit
10.0.1.49 - - [02/Jan/2025 23:18:20] "GET / HTTP/1.1" 200 -
ec2-user@ip-10-0-2-132:~
[ec2-user@ip-10-0-2-132 ~]$ sudo python3 app.py
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:80
* Running on http://10.0.2.132:80
Press CTRL+C to quit
10.0.1.49 - - [02/Jan/2025 23:18:19] "GET / HTTP/1.1" 200 -
Flask App is running from instance 2!

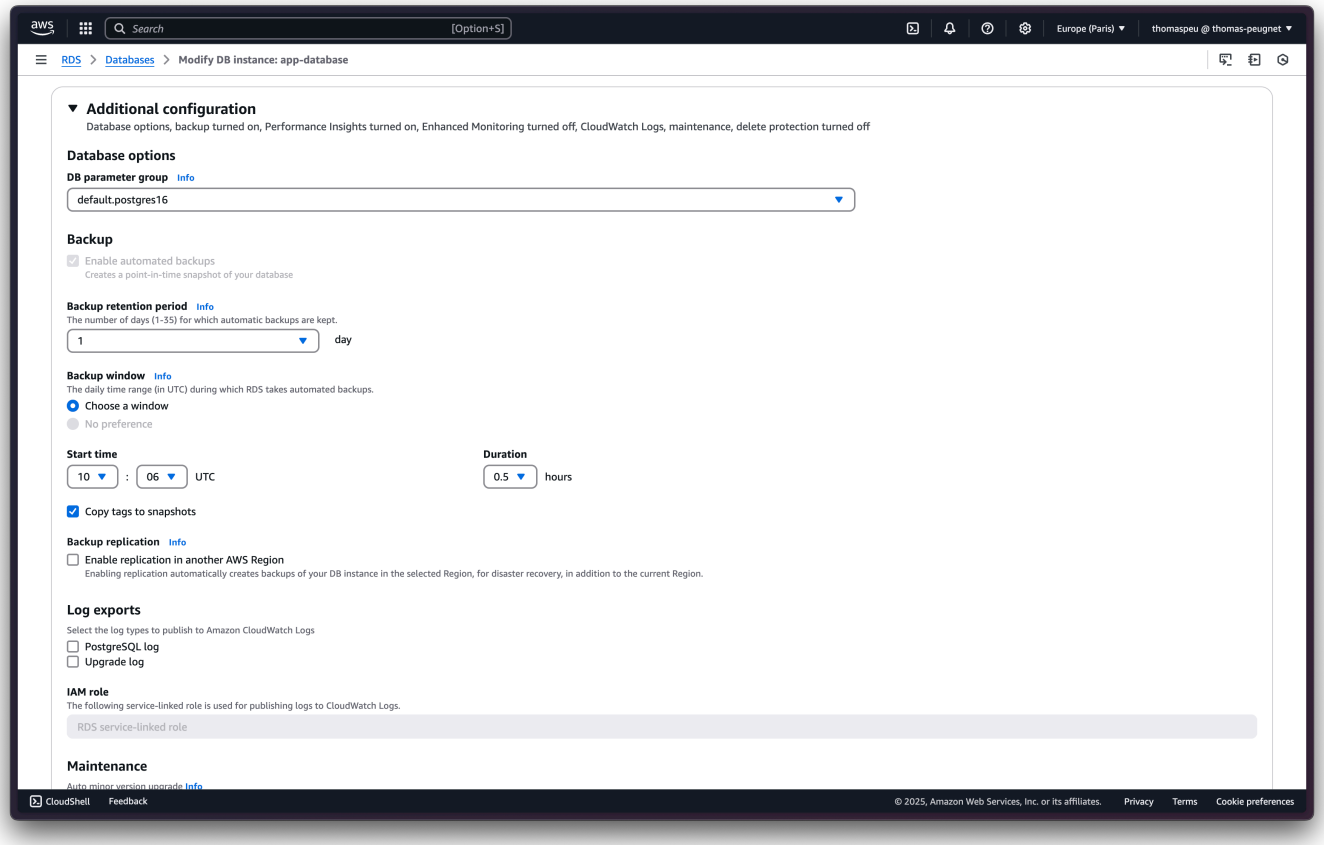

Arc File Edit View Spaces Tabs Archive Extensions Window Help
Ven. 3 janv. à 00:19
ec2-user@ip-10-0-2-155:~
(.venv) [ec2-user@ip-10-0-2-155 ~]$ sudo python3 app.py
* Tip: There are .env files present. Install python-dotenv to use them.
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:80
* Running on http://10.0.2.155:80
Press CTRL+C to quit
10.0.1.49 - - [02/Jan/2025 23:18:20] "GET / HTTP/1.1" 200 -
10.0.1.49 - - [02/Jan/2025 23:18:30] "GET / HTTP/1.1" 200 -
10.0.1.49 - - [02/Jan/2025 23:18:40] "GET / HTTP/1.1" 200 -
10.0.1.49 - - [02/Jan/2025 23:18:50] "GET / HTTP/1.1" 200 -
10.0.1.49 - - [02/Jan/2025 23:19:00] "GET / HTTP/1.1" 200 -
86.254.47.168 - - [02/Jan/2025 23:19:00] "GET / HTTP/1.1" 200 -
ec2-user@ip-10-0-2-132:~
[ec2-user@ip-10-0-2-132 ~]$ sudo python3 app.py
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:80
* Running on http://10.0.2.132:80
Press CTRL+C to quit
10.0.1.49 - - [02/Jan/2025 23:18:19] "GET / HTTP/1.1" 200 -
10.0.1.49 - - [02/Jan/2025 23:18:29] "GET / HTTP/1.1" 200 -
10.0.1.49 - - [02/Jan/2025 23:18:39] "GET / HTTP/1.1" 200 -
10.0.1.49 - - [02/Jan/2025 23:18:49] "GET / HTTP/1.1" 200 -
10.0.1.49 - - [02/Jan/2025 23:18:59] "GET / HTTP/1.1" 200 -
Flask App is running!

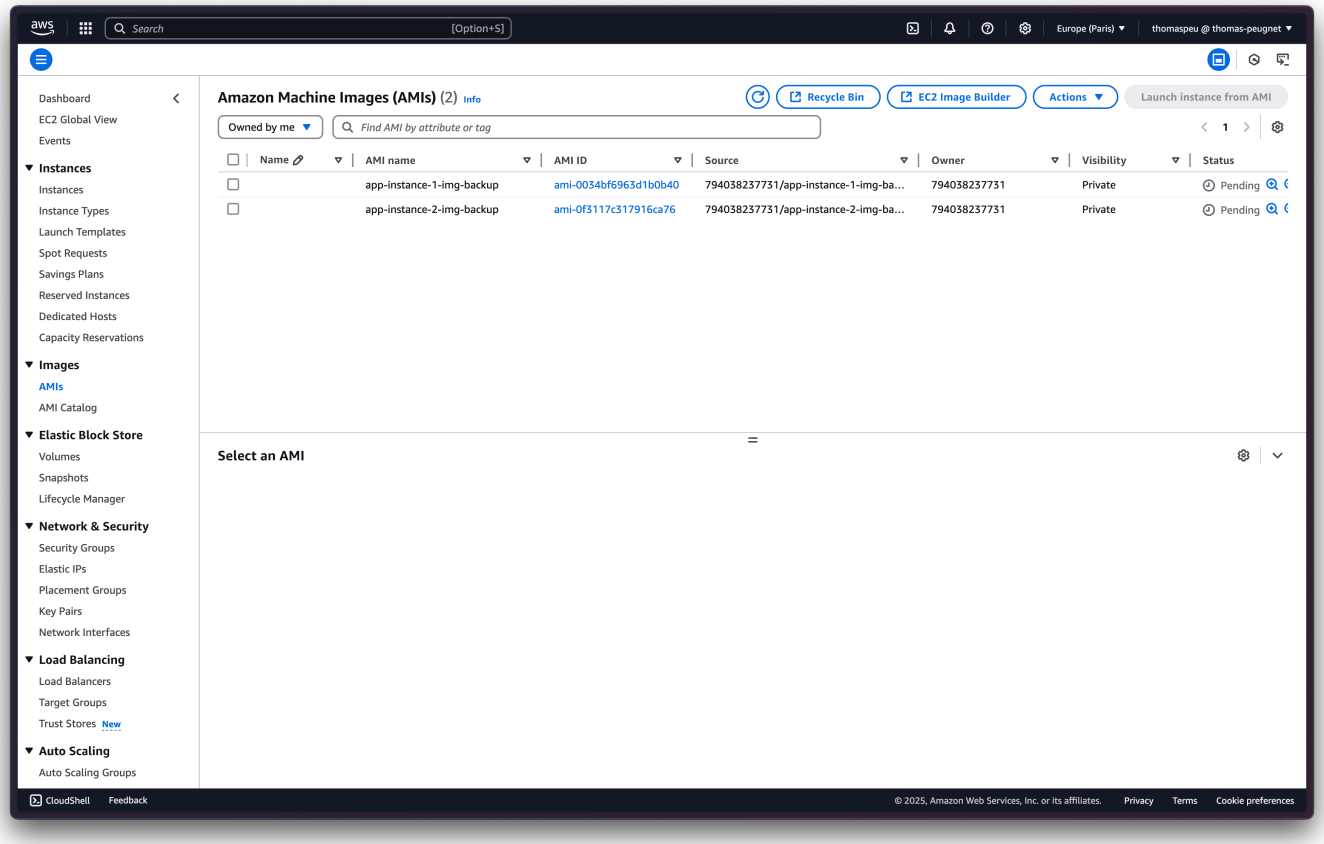
Configurations Annexes

Sauvegardes

Nous configurons les sauvegardes automatiques pour notre base de données et nous sauvegardons nos images AMIs.

Note: Nous considérons ici, bien que ce soit le sujet d'une amélioration future, que le code de l'application Flask est déployé via une pipeline CI/CD. Il n'est donc nécessaire de faire une sauvegarde de notre instance que pour en accélérer le déploiement, et non, à terme, pour en sauvegarder le code.

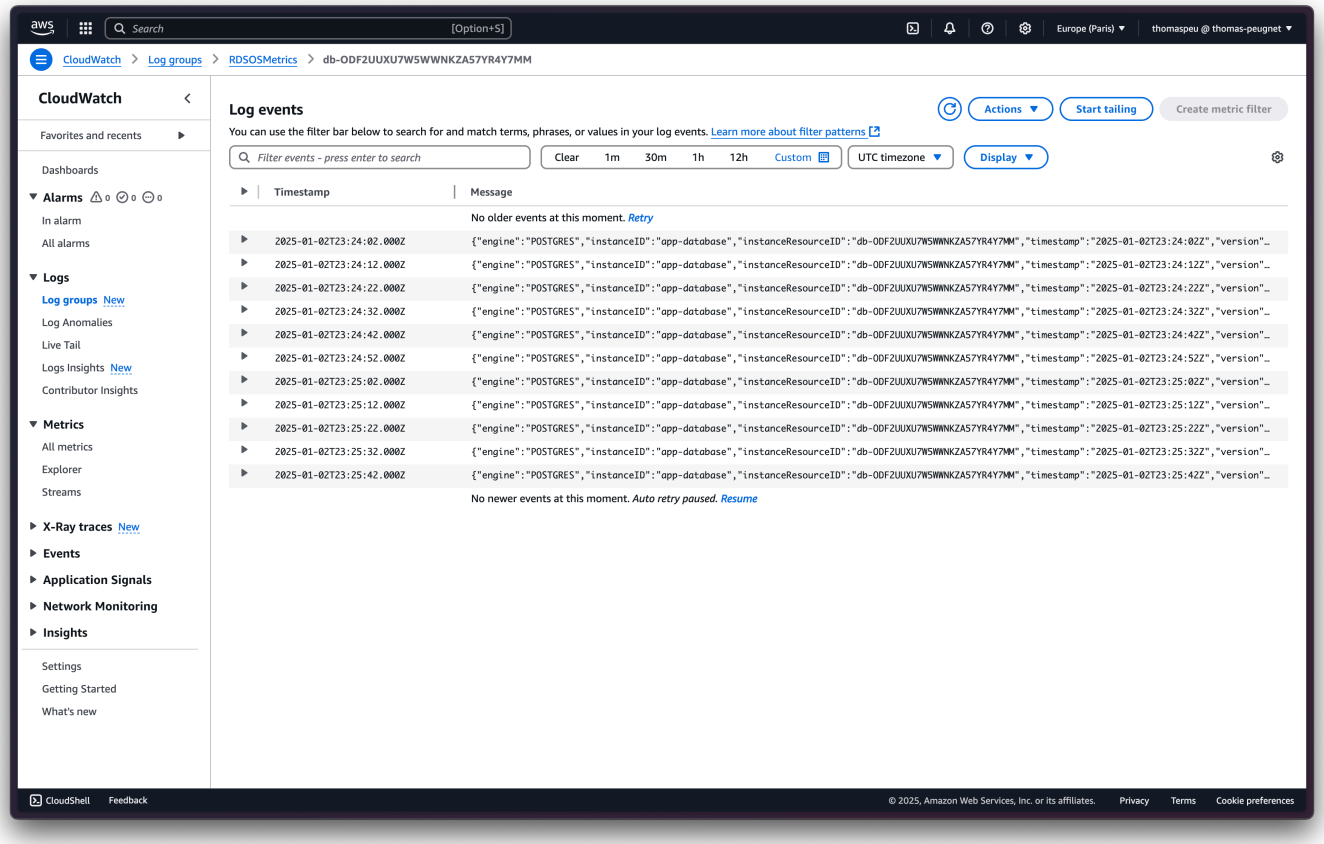


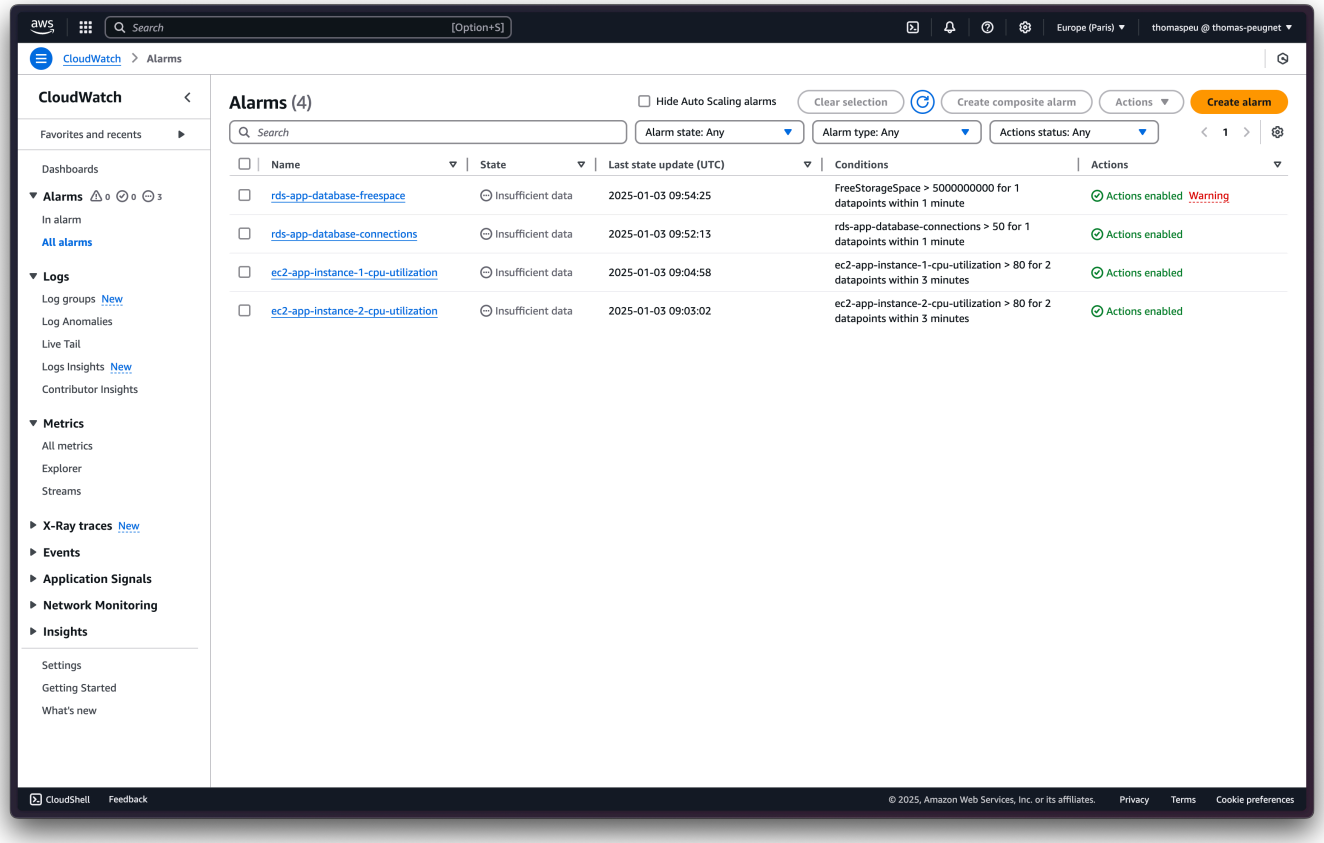


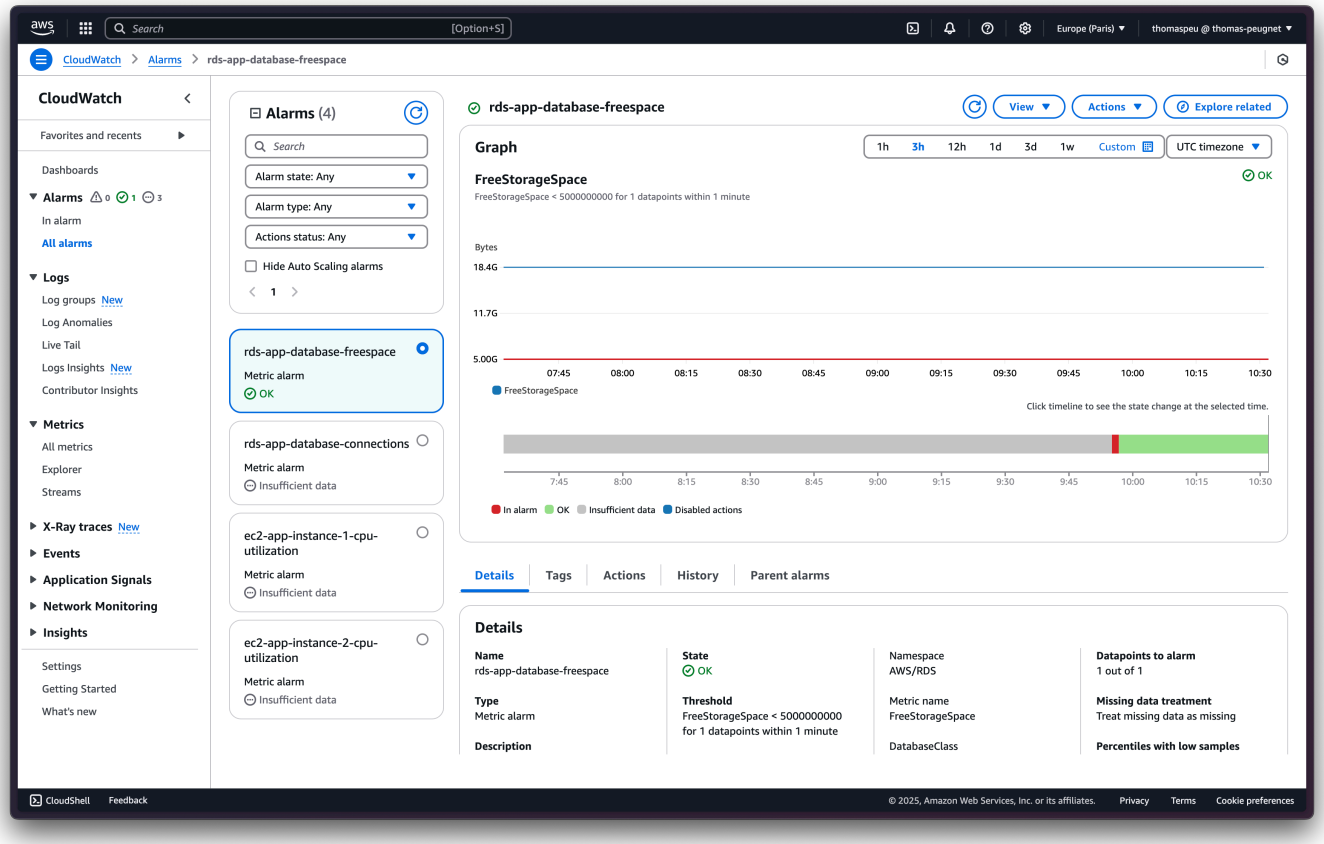
Supervision

Nous configurons la supervision de RDS avec CloudWatch et créons des alarmes pour le CPU de nos instances de serveurs web et de base de données.

Nous constatons que nos logs sont bien visibles depuis CloudWatch.


Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
CloudWatch > Log groups > RDSOSMetrics > db-ODF2UUXU7W5WWNKZA57YR4Y7MM
CloudWatch
Favorites and recents
Dashboards
Alarms 0 0 0
In alarm
All alarms
Logs
Log groups New
Log Anomalies
Live Tail
Logs Insights New
Contributor Insights
Metrics
All metrics
Explorer
Streams
X-Ray traces New
Events
Application Signals
Network Monitoring
Insights
Settings
Getting Started
What's new
Log events
Actions
Start tailing
Create metric filter
You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns
Filter events - press enter to search
Clear 1m 30m 1h 12h Custom
UTC timezone
Display
Timestamp
Message
No older events at this moment. Retry
2025-01-02T23:24:02.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:24:02Z","version"…
2025-01-02T23:24:12.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:24:12Z","version"…
2025-01-02T23:24:22.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:24:22Z","version"…
2025-01-02T23:24:32.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:24:32Z","version"…
2025-01-02T23:24:42.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:24:42Z","version"…
2025-01-02T23:24:52.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:24:52Z","version"…
2025-01-02T23:25:02.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:25:02Z","version"…
2025-01-02T23:25:12.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:25:12Z","version"…
2025-01-02T23:25:22.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:25:22Z","version"…
2025-01-02T23:25:32.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:25:32Z","version"…
2025-01-02T23:25:42.000Z {"engine":"POSTGRES","instanceID":"app-database","instanceResourceID":"db-ODF2UUXU7W5WWNKZA57YR4Y7MM","timestamp":"2025-01-02T23:25:42Z","version"…
No newer events at this moment. Auto retry paused. Resume
CloudShell Feedback
© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences


Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
CloudWatch > Alarms
CloudWatch
Favorites and recents
Dashboards
Alarms 0 0 3
In alarm
All alarms
Logs
Log groups New
Log Anomalies
Live Tail
Logs Insights New
Contributor Insights
Metrics
All metrics
Explorer
Streams
X-Ray traces New
Events
Application Signals
Network Monitoring
Insights
Settings
Getting Started
What's new
Alarms (4)
Hide Auto Scaling alarms
Clear selection
Create composite alarm
Actions
Create alarm
Search
Alarm state: Any
Alarm type: Any
Actions status: Any
1
Name
State
Last state update (UTC)
Conditions
Actions
rds-app-database-freespace Insufficient data 2025-01-03 09:54:25 FreeStorageSpace > 5000000000 for 1 datapoints within 1 minute Actions enabled Warning
rds-app-database-connections Insufficient data 2025-01-03 09:52:13 rds-app-database-connections > 50 for 1 datapoints within 1 minute Actions enabled
ec2-app-instance-1-cpu-utilization Insufficient data 2025-01-03 09:04:58 ec2-app-instance-1-cpu-utilization > 80 for 2 datapoints within 3 minutes Actions enabled
ec2-app-instance-2-cpu-utilization Insufficient data 2025-01-03 09:03:02 ec2-app-instance-2-cpu-utilization > 80 for 2 datapoints within 3 minutes Actions enabled
CloudShell Feedback
© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



Note: L'alarme ci-dessus (freespace) a été en status `in Alarm` car nous nous sommes trompés dans le threshold, nous avions indiqué que la notification devait partir si l'espace disponible était supérieur (et non inférieur) à 5.000.000.000 bytes (5GB). Cela a été corrigé, d'où le statut de l'alarme mis à jour en `OK`.

En résumé, nous avons créé les alarmes suivantes:

Alarme CPU pour EC2 (2 alarmes) :

- Instances `app-instance-1` et `app-instance-2` : Déclenchées lorsque l'utilisation CPU dépasse 80 % pendant une période prolongée. Elles servent à détecter une surcharge de traitement sur les instances.

Alarme sur les connexions actives pour RDS :

- Déclenchée lorsque le nombre de connexions simultanées à la base de données dépasse un seuil défini à 20 connexions. Elle permet de surveiller les pics de trafic ou de charge sur la base.

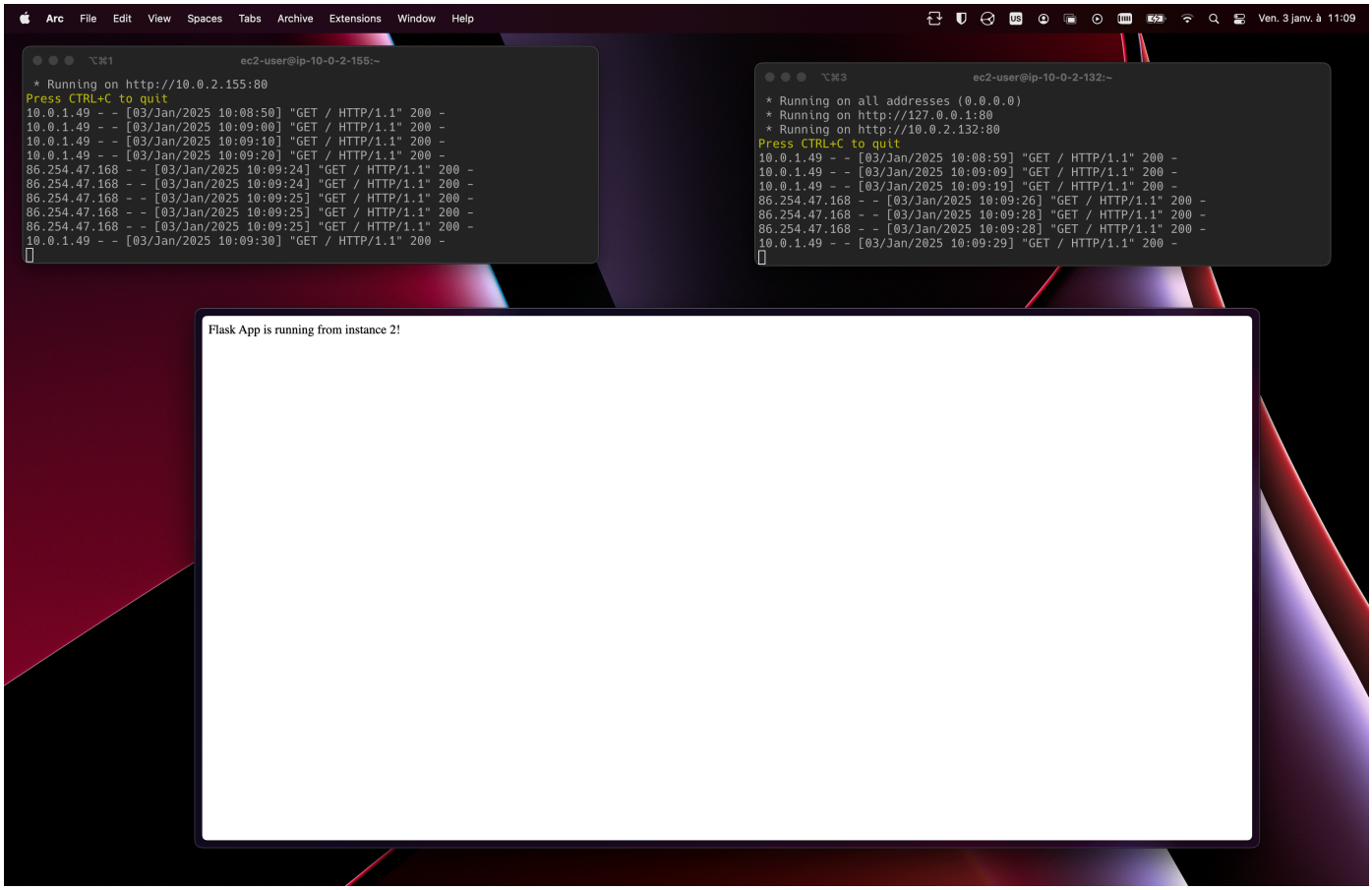
Alarme sur l'espace disque disponible pour RDS :

- Déclenchée lorsque l'espace disque disponible sur la base de données descend en dessous de 5GB. Elle sert à prévenir une saturation du stockage.

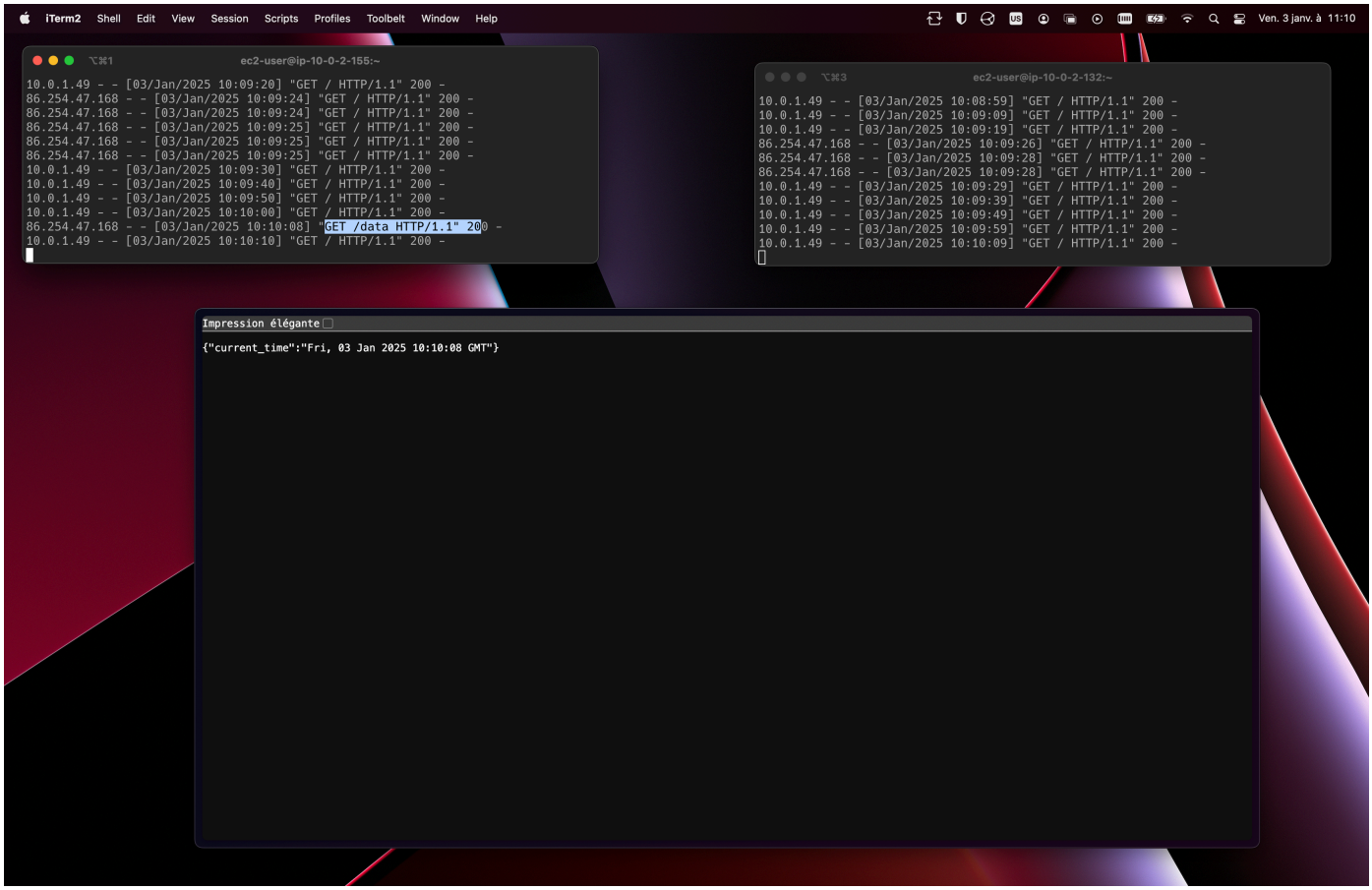
Tests

Nous testons l'accès à notre application via un navigateur web.

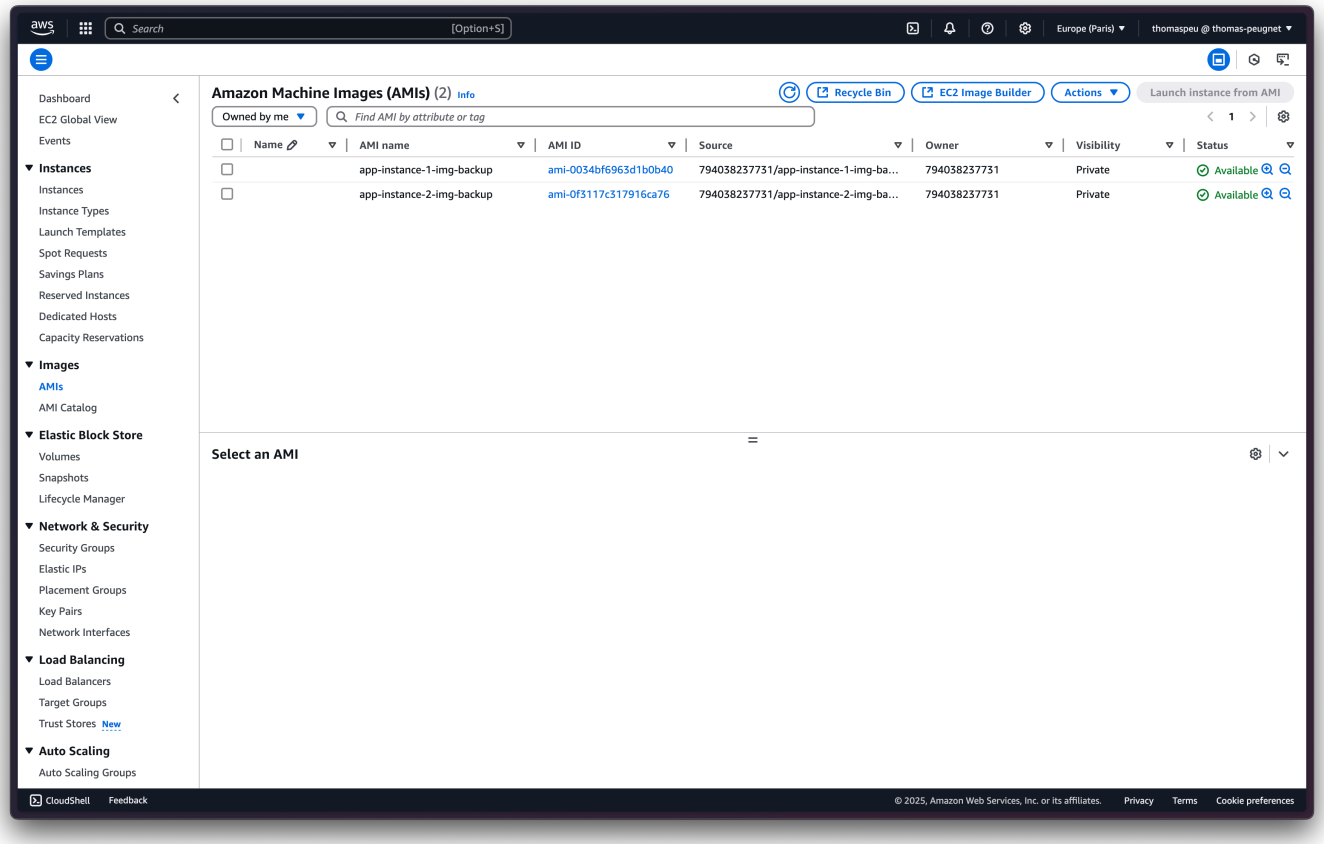
Cela semble bien fonctionner pour la route `/` en utilisant le nom DNS de notre loadbalancer.



Il en est de même pour la route `/data`.

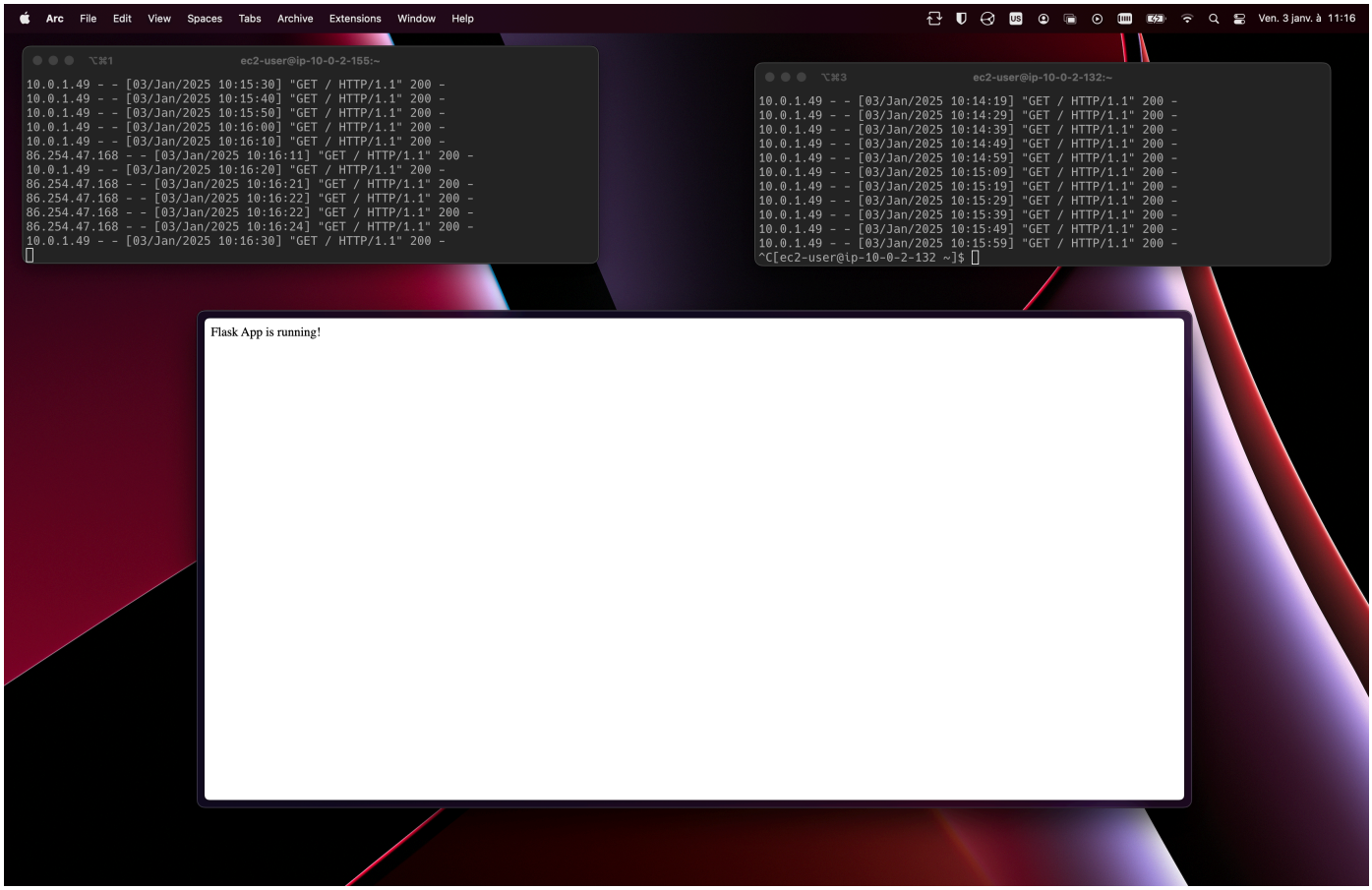


Nous testons les sauvegardes, en vérifiant que les images AMIs ont bien été créés à notre demande.



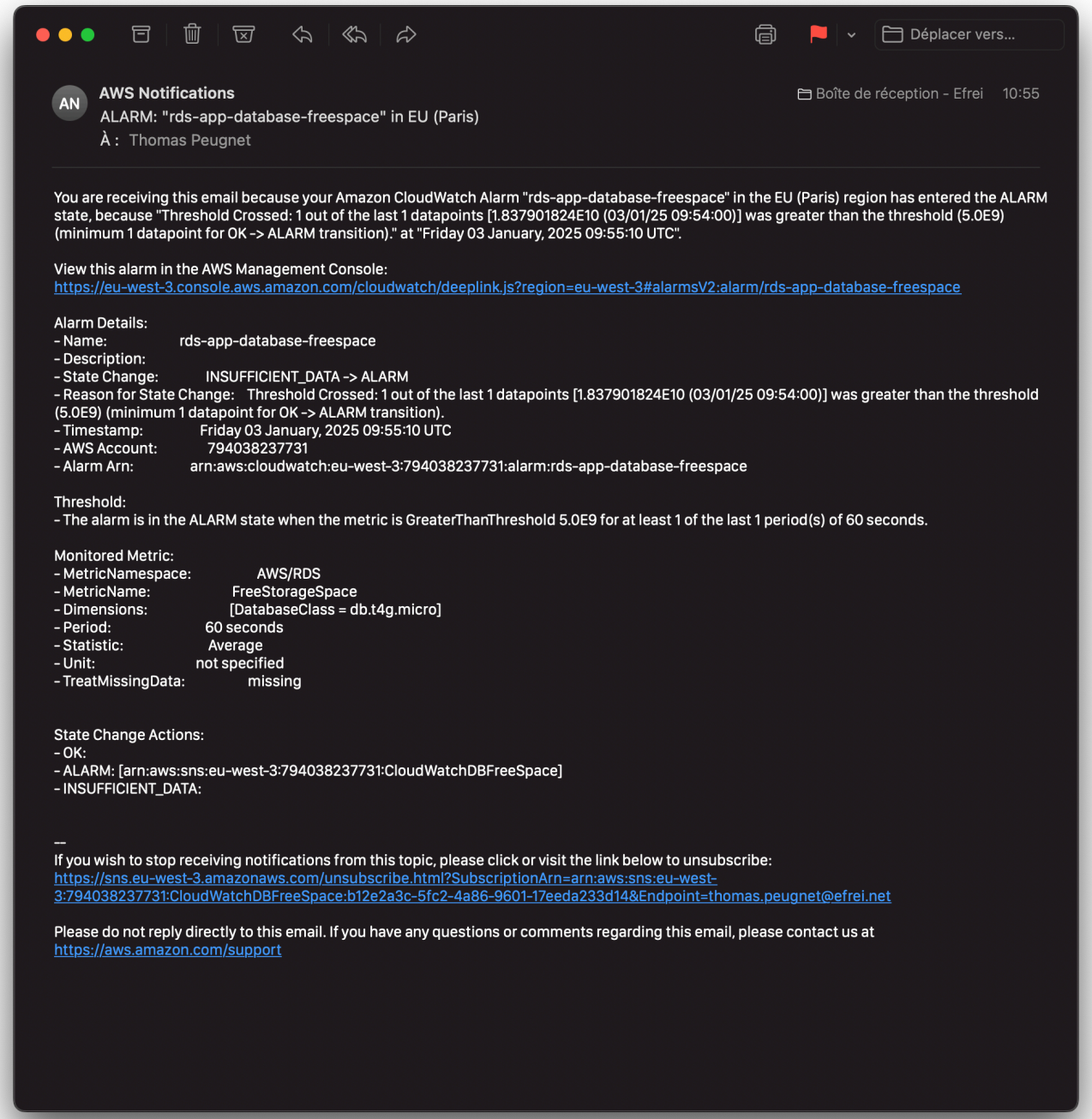
Nous testons la résilience en éteignant une instance et en vérifiant que celle toujours up est bien utilisée pour servir notre application.

Nous constatons que seule la première instance du serveur Flask est utilisée pour répondre, cela fonctionne donc correctement.

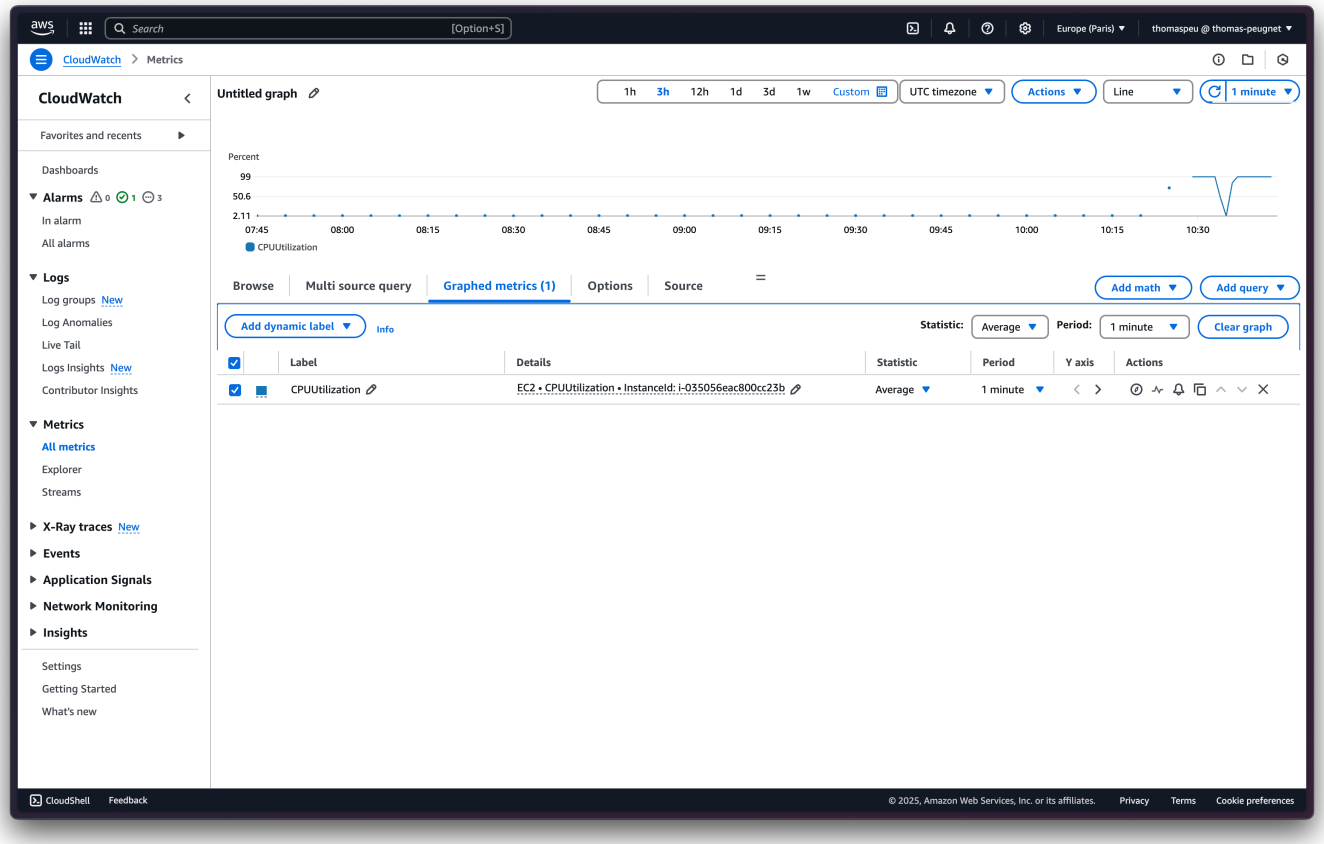


Nous testons l'une de nos alarmes (la plus rapide à mettre en place) pour valider la bonne réception des notification par mail.

Après avoir fait notre erreur de threshold sur l'espace disponible sur la base de données, nous avons rapidement reçu le mail suivant. Les notifications semblent donc fonctionner correctement.



Nous utilisons le package `stress` pour faire un stress test des instances. Nous pouvons constater au niveau du monitoring que les graphs sont tout à fait cohérents.



Nous utilisons la commande `stress --cpu 1 --timeout 10000`.

Les alarmes de CPU ne sont pas encore prêtes à être utilisées par manque de temps (elles demeurent en statut `Insuficien Data`), mais fonctionneront de façon similaire à l'alerte Freespace éprouvée juste auparavant.

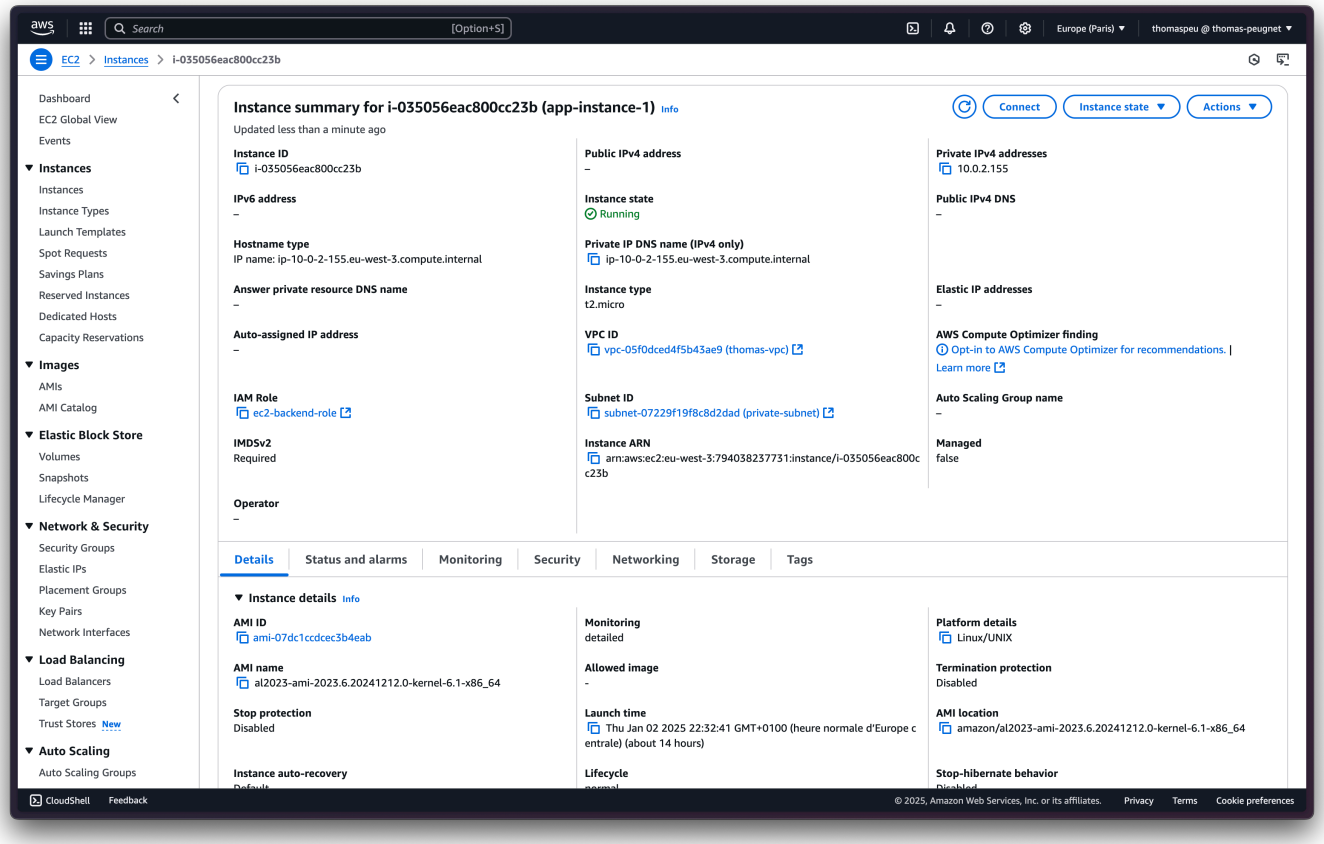
IAM

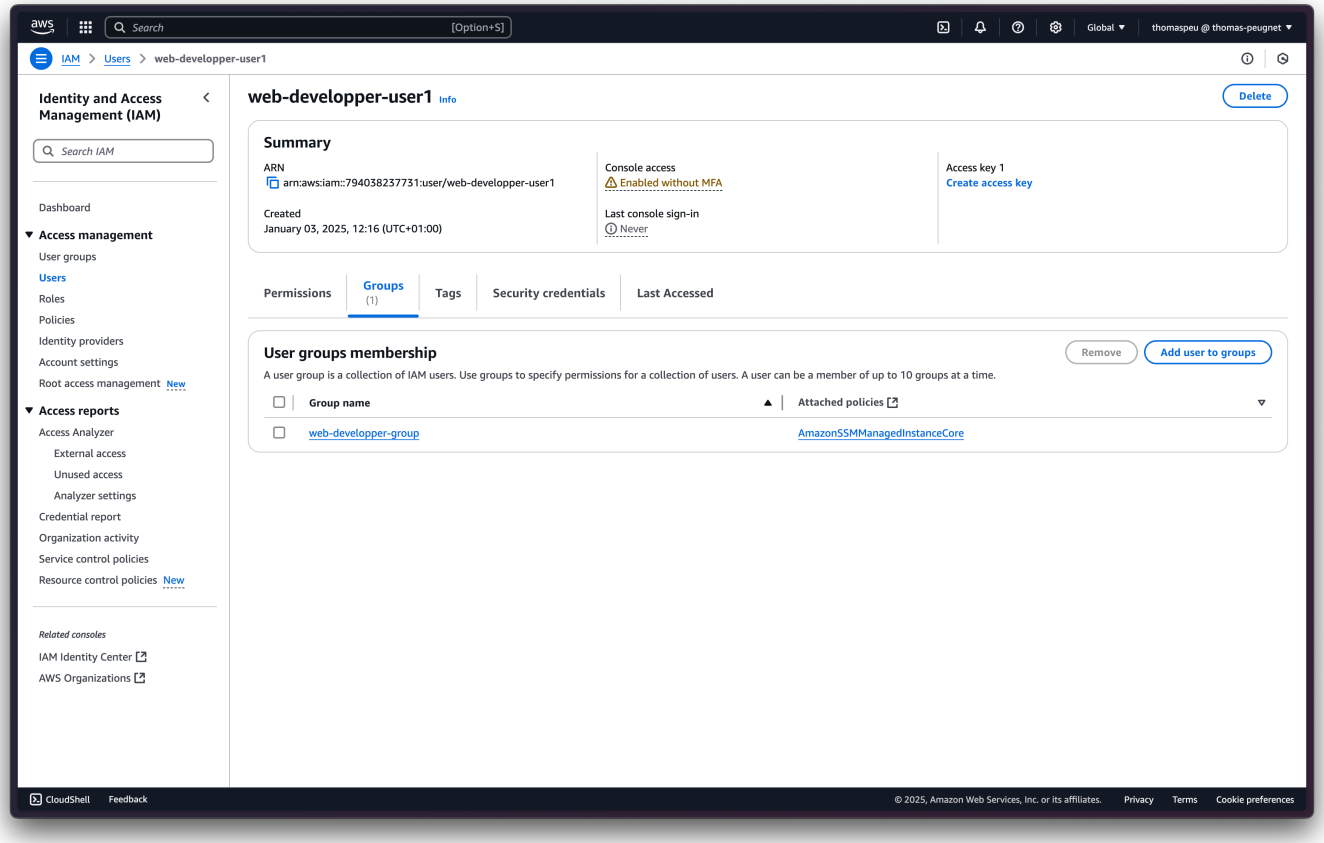
Nous créons un rôle `ec2-backend-role` avec la permission AmazonSSMManagedInstanceCore pour gérer les instances EC2 dans le subnet privé via Systems Manager.

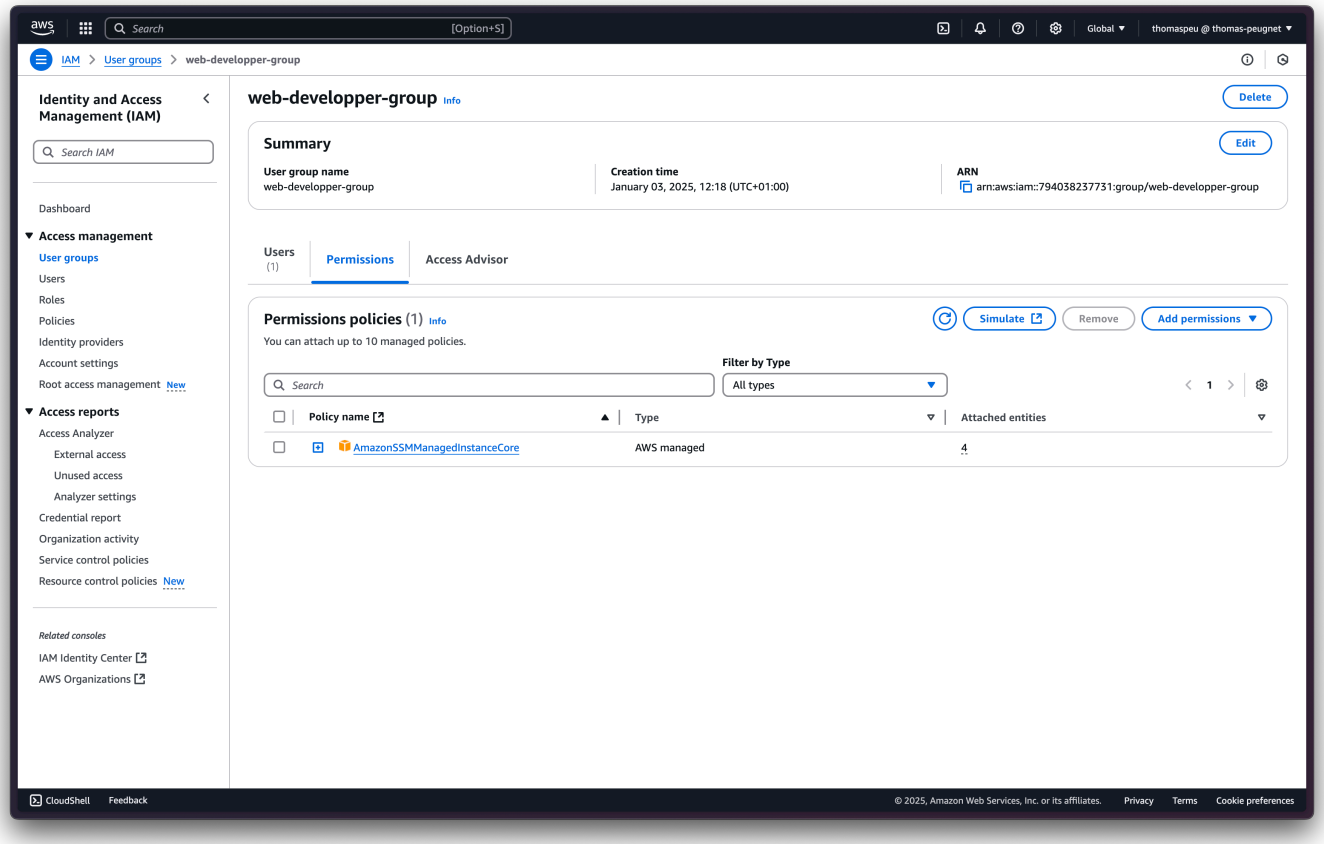
Nous créons ensuite l'utilisateur `web-developper-user1` et le groupe `web-developper-group`, auquel nous attachons également la permission AmazonSSMManagedInstanceCore.

L'utilisateur est ajouté au groupe et une clé d'accès est générée pour des connexions AWS CLI.

Nous associons nos instances EC2 contenant l'application Flask à notre rôle `ec2-backend-role`. Nous avons testé la connexion via un autre utilisateur et cela fonctionne parfaitement.


Search [Option+S]
Europe (Paris)
thomaspeu @ thomas-peugnet
EC2 > Instances > i-035056eac800cc23b
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Target Groups
Trust Stores New
Auto Scaling
Auto Scaling Groups
Instance summary for i-035056eac800cc23b (app-instance-1) Info
Updated less than a minute ago
Connect
Instance state
Actions
Instance ID
i-035056eac800cc23b
IPv6 address
–
Hostname type
IP name: ip-10-0-2-155.eu-west-3.compute.internal
Answer private resource DNS name
–
Auto-assigned IP address
–
IAM Role
ec2-backend-role
IMDSv2
Required
Operator
–
Public IPv4 address
–
Instance state
Running
Private IP DNS name (IPv4 only)
ip-10-0-2-155.eu-west-3.compute.internal
Instance type
t2.micro
VPC ID
vpc-05f0dced4f5b43ae9 (thomas-vpc)
Subnet ID
subnet-07229f19f8c8d2dad (private-subnet)
Instance ARN
arn:aws:ec2:eu-west-3:794038237731:instance/i-035056eac800cc23b
Private IPv4 addresses
10.0.2.155
Public IPv4 DNS
–
Elastic IP addresses
–
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
Learn more
Auto Scaling Group name
–
Managed
false
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Instance details Info
AMI ID
ami-07dc1ccdcec3b4eab
AMI name
al2023-ami-2023.6.20241212.0-kernel-6.1-x86_64
Stop protection
Disabled
Instance auto-recovery
Monitoring
detailed
Allowed image
-
Launch time
Thu Jan 02 2025 22:32:41 GMT+0100 (heure normale d'Europe centrale) (about 14 hours)
Lifecycle
Platform details
Linux/UNIX
Termination protection
Disabled
AMI location
amazon/al2023-ami-2023.6.20241212.0-kernel-6.1-x86_64
Stop-hibernate behavior
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

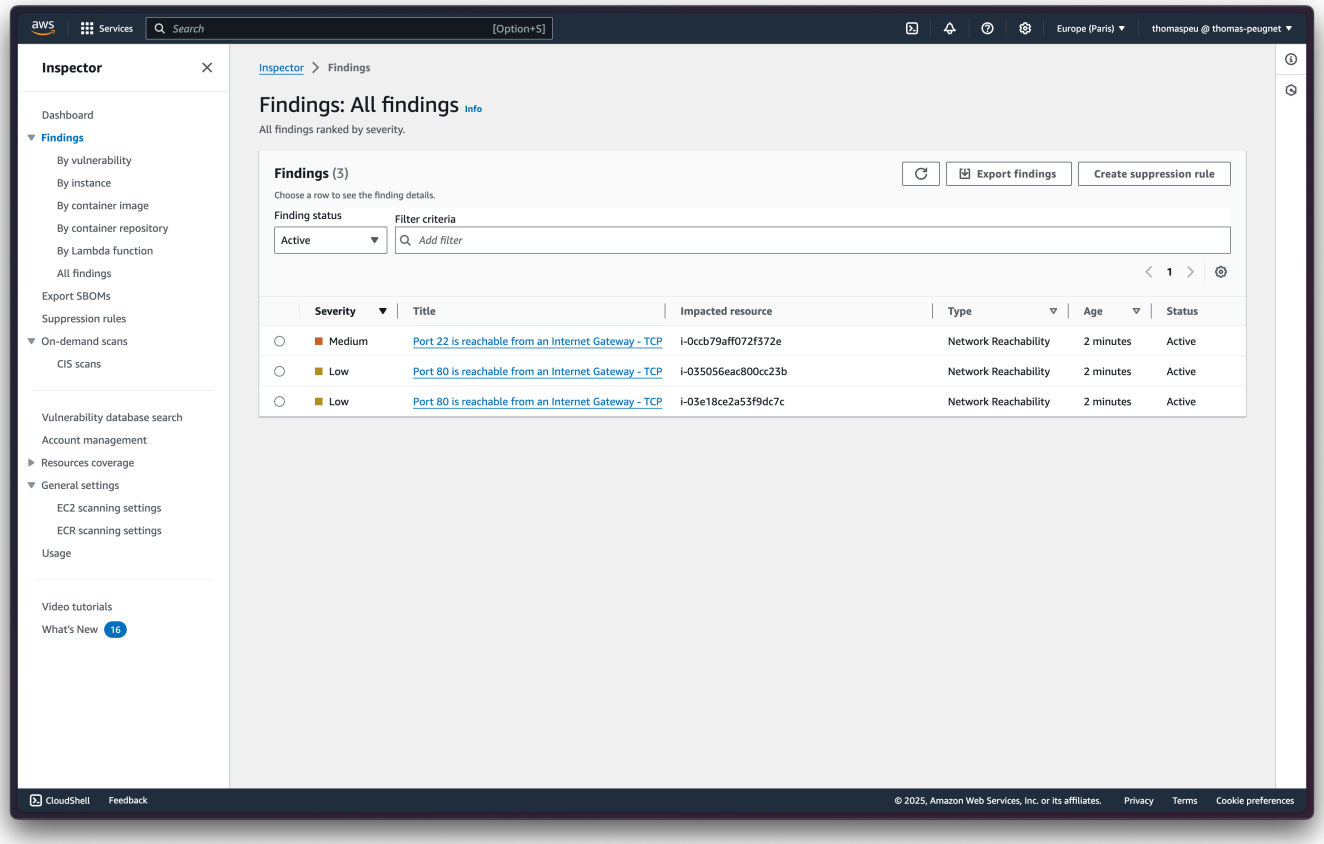

Search [Option+S]
Global
thomaspeu @ thomas-peugnet
IAM > Users > web-developper-user1
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management New
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Credential report
Organization activity
Service control policies
Resource control policies New
Related consoles
IAM Identity Center
AWS Organizations
web-developper-user1 Info
Delete
Summary
ARN
arn:aws:iam::794038237731:user/web-developper-user1
Created
January 03, 2025, 12:16 (UTC+01:00)
Console access
Enabled without MFA
Last console sign-in
Never
Access key 1
Create access key
Permissions
Groups (1)
Tags
Security credentials
Last Accessed
User groups membership
Remove
Add user to groups
A user group is a collection of IAM users. Use groups to specify permissions for a collection of users. A user can be a member of up to 10 groups at a time.
Group name
Attached policies
web-developper-group
AmazonSSMManagedInstanceCore
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences



AWS Inspector

Nous activons AWS Inspector pour analyser les vulnérabilités de nos ressources.

Nous obtenons le résultat suivant, tout à fait cohérent avec la structure de notre VPC.



Nos 2 instances dans notre subnet privé sont bel et bien accessibles en HTTP via le Loadbalancer sur le port 80. Notre instance dans notre subnet public (servant de rebond) est bien exposée publiquement sur le port 22.

Améliorations

Réseau (VPC & Subnets)

- Supprimer l'accès public direct à l'instance de rebond sur le port 22 (Systems Manager Session Manager)
- Utiliser un WAF (Web Application Firewall) pour protéger l'application Flask

Instances EC2

- Éliminer l'utilisation directe de clés SSH (utiliser AWS Systems Manager)
- Configurer des snapshots automatiques et réguliers via un Lifecycle Manager pour les volumes attachés aux instances.

Déploiement du code

- Mise en place d'une CI/CD avec GitHub Actions (ou équivalent GitLab) pour le déploiement de l'application Flask
- Passer l'application Flask à une image Docker et utiliser un registry (EKS par exemple)

Sécurité

- Automatiser les mises à jour logicielles des instances EC2 (AWS Systems Manager Patch Manager)
- Centraliser les logs avec AWS CloudWatch Logs Insights pour analyser les comportements considérés comme anormaux

Alarmes

- Ajouter d'autres alarmes pour surveiller encore davantage la santé des instances et des applications :
  - Memory Utilization
  - HTTP 4xx/5xx sur le Load Balancer

IAM

- Granularité des permissions : Créer des rôles spécifiques pour chaque type d'instance ou service. Dans notre cas, un rôle pour la gestion de base de données pourrait être fort intéressant.
- Surveillance des accès : Mettre en place des alertes via CloudWatch et AWS CloudTrail pour détecter les accès considérés comme anormaux ou non autorisés.
- Multi-Factor Authentication (MFA) : Activer le MFA pour tous les utilisateurs IAM ayant accès à des ressources critiques.

Suppression

Nous procédons à la suppression des ressources dans l'ordre suivant :

1. Nous supprimons les alarmes pour les métriques des instances EC2 et de la base de données RDS.
2. Nous supprimons les sauvegardes, y compris les snapshots RDS et les AMIs EC2.
3. Nous supprimons la base de données RDS et le DB Subnet Group associé.
4. Nous supprimons le Network Load Balancer et le Target Group.
5. Nous terminons les instances EC2, incluant celles du subnet privé et l'instance de rebond.
6. Nous supprimons les Security Groups créés pour RDS, EC2, et le Load Balancer.
7. Nous supprimons la NAT Gateway et libérons l'Elastic IP associée.
8. Nous supprimons le VPC et toutes ses ressources associées (Internet Gateway, subnets, tables de routage).
9. Nous supprimons les rôles, groupes, et utilisateurs IAM créés pour le projet.
10. Nous utilisons AWS Resource Groups Tag Editor pour identifier et supprimer toute ressource résiduelle non détectée.