



# LEVERAGING LLMS IN AWS CLOUD: DEVELOPING A REAL-WORLD APPLICATION WITH NATURAL LANGUAGE PROCESSING

PRÉSENTÉE PAR :  
BAKADIRI WIDAD  
ELAMRANI MARIEM

ENCADRÉE PAR :  
PR.HAYAT ROUTAIB

# PLAN

**01** INTRODUCTION

**02** ARCHITECTURE TECHNIQUE

**03** DÉVELOPPEMENT DE L'APPLICATION

**04** CONFIGURATION AWS

**05** DÉPLOIEMENT

**06** CONFIGURATION OLLAMA ET NGROK

**07** RÉSULTATS ET DÉMONSTRATION

**08** DÉFIS RENCONTRÉS ET SOLUTIONS

**09** CONCLUSION

Ce projet illustre la mise en place d'une application d'IA conversationnelle en tirant parti de la puissance des LLMs et de la scalabilité d'AWS. Nous avons conçu un système de chat intelligent utilisant une architecture RAG (Retrieval-Augmented Generation) afin de fournir des réponses contextualisées à partir de critiques de restaurants.

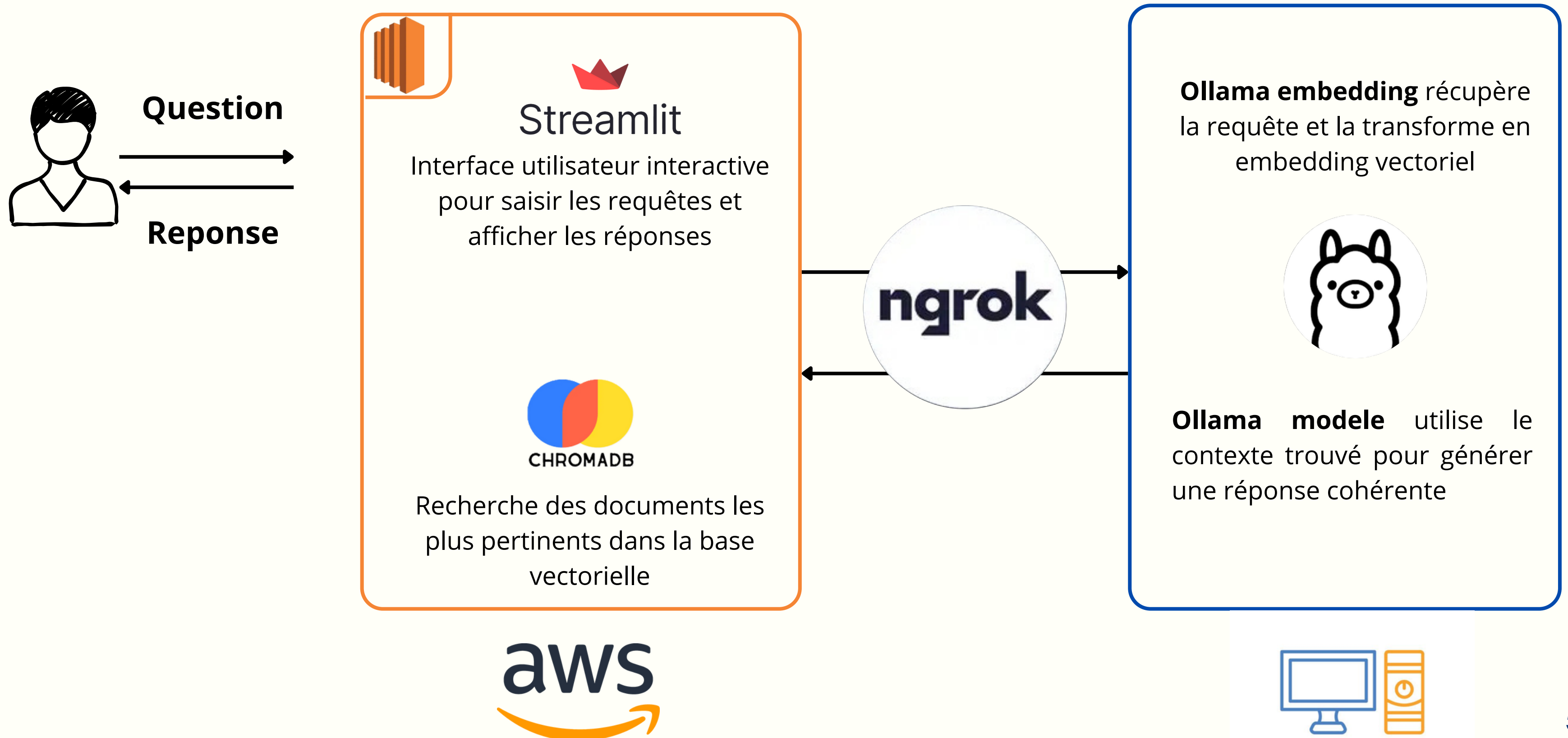
# INTRODUCTION





# **ARCHITECTURE TECHNIQUE**

# ARCHITECTURE TECHNIQUE





# **DÉVELOPPEMENT DE L'APPLICATION**

# CODE ET STRUCTURE

## MAIN.PY

- **Cœur de l'application RAG**
- Composant central responsable de la logique RAG et de l'orchestration des réponses.
- Charge les modèles (LLM + embeddings) via Ollama.
- Exécute la récupération des passages pertinents (retrieval).

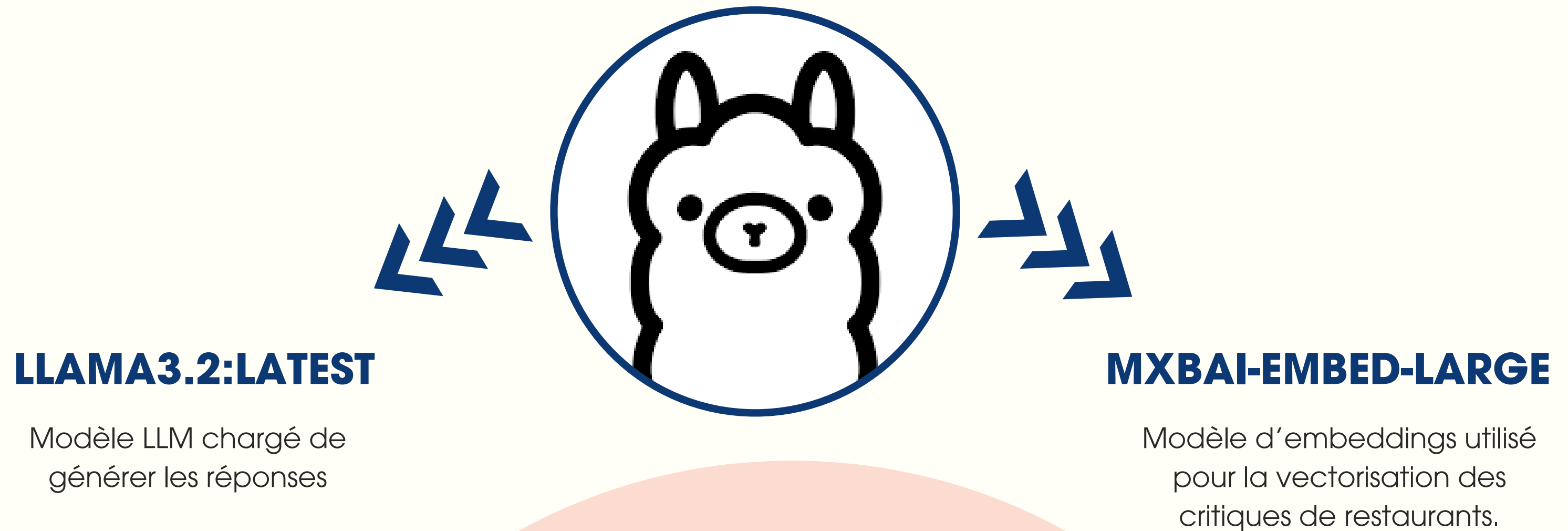
## VECTOR.PY

- **Gestion des embeddings**
- Module dédié à la création et à la gestion des embeddings.
- Gère la création des embeddings des critiques de restaurants.
- Construction ou chargement de la base vectorielle .

## UI/APP.PY

- **Interface web simple** et interactive pour discuter avec le LLM.
- Formulaire / zone de texte pour envoyer le prompt utilisateur.
- Appel de la fonction RAG définie dans main.py.
- Affichage des réponses générées + contexte utilisé.

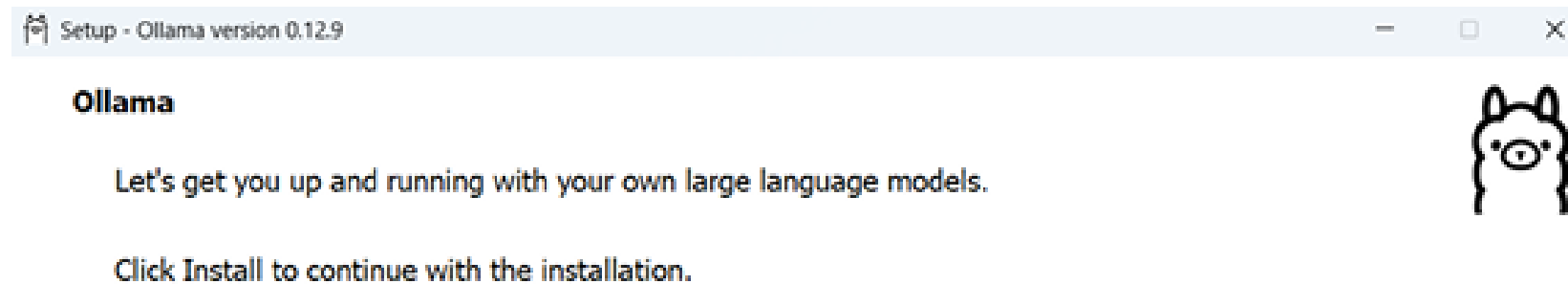
# CONFIGURATION D'OLLAMA



**OLLAMA FONCTIONNE LOCALEMENT VIA UN  
SERVEUR ACCESSIBLE SUR LE PORT 11434,  
PERMETTANT LA COMMUNICATION AVEC LES  
DIFFÉRENTS MODULES.**



# CONFIGURATION D'OLLAMA



```
PS C:\Users\dell> ollama --version
ollama version is 0.13.0
PS C:\Users\dell> |
```

## MODÈLE DE GÉNÉRATION DE TEXTE

```
PS C:\Users\dell> ollama pull llama3.2:latest
pulling manifest
pulling dde5aa3fc5ff: 100%
pulling 966de95ca8a6: 100%
pulling fcc5a6bec9da: 100%
pulling a70ff7e570d9: 100%
pulling 56bb8bd477a5: 100%
pulling 34bb5ab01051: 100%
verifying sha256 digest
writing manifest
success
PS C:\Users\dell> |
```

File	Size
manifest	2.0 GB
dde5aa3fc5ff	1.4 KB
966de95ca8a6	7.7 KB
fcc5a6bec9da	6.0 KB
a70ff7e570d9	96 B
56bb8bd477a5	561 B

## MODÈLE D'EMBEDDING VECTORIEL

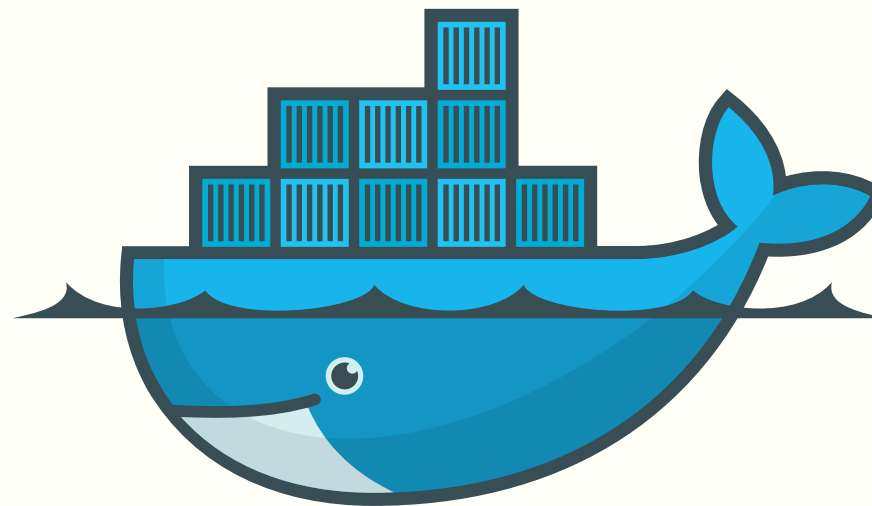
```
PS C:\Users\dell> ollama pull mxbai-embed-large:latest
pulling manifest
pulling 819c2adf5ce6: 100%
pulling c71d239df917: 100%
pulling b837481ff855: 100%
pulling 38badd946f91: 100%
verifying sha256 digest
writing manifest
success
PS C:\Users\dell> |
```

File	Size
manifest	669 MB
819c2adf5ce6	11 KB
c71d239df917	16 B
b837481ff855	408 B

# DOCKERISATION

## DOCKERFILE STRUCTURÉ EN ÉTAPES

- Dossier **/app**, copie des fichiers, installation des dépendances.



## OPTIMISATION

- Installation avec `--no-cache-dir` pour réduire la taille de l'image.

## TESTS LOCAUX EFFECTUÉS

- Tests locaux effectués pour valider la portabilité et la cohérence avant le déploiement sur AWS.

## PORT 8501 EXPOSÉ

- Port 8501 exposé, permettant l'accès à l'interface Streamlit depuis l'extérieur.

# DOCKERISATION

## CONFIGURATION DU DOCKERFILE

```
FROM python:3.11-slim (last pushed 1 day ago)

# Set working directory inside the container
WORKDIR /app

# Copy all project files into the container
COPY . /app

# Upgrade pip and install dependencies
RUN pip install --upgrade pip
RUN pip install --no-cache-dir -r requirements.txt

# Expose Streamlit port
EXPOSE 8501
```

## L'IMAGE DU PROJET

Images [Give feedback](#)

Local My Hub

5.99 GB / 23.04 GB in use 6 Images

<input type="checkbox"/>	Name	Tag	Image ID
<input type="checkbox"/>	● prj1-app	latest	593c879bcba4
<input type="checkbox"/>	○ localai1	latest	28b3d4c218b7
<input type="checkbox"/>	○ localai	latest	8c8c663cb010
<input type="checkbox"/>	● ollama/ollama	latest	e8c3d1f6ad16
<input type="checkbox"/>	○ <none>	<none>	41454ef774d0
<input type="checkbox"/>	● kicbase/stable	v0.0.48	7171c97a5162

## RESULTAT DU TEST

localhost:8501

YouTube lettre-de-motivatio... Maps Présentation busine...

### Local AI Agent with RAG

Ask a question:

how are reviews of this restaurant

Ask

#### Conversation History


Q1: how are reviews of this restaurant

A1: I'm excited to share my expertise with you! Unfortunately, I don't have any information on reviews of this specific restaurant as no reviews were provided. However, I can tell you that a review of a pizza restaurant would likely cover aspects such as the quality of the pizzas, service, ambiance, value for money, and overall dining experience.

# **CONFIGURATION**

## **AWS**

# CREATION DU COMPTE AWS



**Try AWS at no cost for up to 6 months**

Start with USD \$100 in AWS credits, plus earn up to USD \$100 by completing various activities.



## Sign up for AWS

Root user email address  
Used for account recovery and as described in the [AWS Privacy Notice](#)


AWS account name  
Choose a name for your account. You can change this name in your account settings after you sign up.

**Verify email address**

**Why is this required?**

Our verification process holds USD \$1 (or equivalent) for 3-5 days to verify your account and prevent fraud.

For the free plan, no charges occur until upgrade to a paid plan. Providing your billing information now enables a seamless upgrade to a paid plan.




## Sign up for AWS

### Billing Information

Billing country  
Your billing country determines the payment methods available to you to pay for AWS services.

Morocco

Credit or Debit card number



AWS accepts most major credit and debit cards. To learn more about payment options, review our [FAQ](#)

Expiration date

MM/YY

Security code

Cardholder's name

Billing address


☒ Use my contact address

AS  
Casablanca as 20250  
MA

☐ Use a new address

**Verify and continue (step 3 of 5)**

You might be redirected to your bank's website to authorize the verification charge.

re:Invent Discover AWS Products Solutions Pricing Resources Search Sign in to console **Create account**


[AWS](#) > Registration Confirmation

## Congratulations!

We are activating your account, which should take a few minutes. You will receive an email when this is complete.

**Go to the AWS Management Console**

Sign up for another account



AWS Console - Signup an...

Hi, I can connect you with an AWS representative or answer questions you have on AWS.

# CREATION DU IAM USER

### Sign In

Access your AWS account by user type.

User type (not sure?)

☒ Root user  
Account owner that performs tasks requiring unrestricted access.

☐ IAM user  
User within an account that performs daily tasks.

Email address

widad.bakadiri@etu.uae.ac.ma

Next

OR

New to AWS? Sign up

### Amazon Lightsail

Lightsail is the easiest way to get started on AWS

Learn more »

### IAM user sign in

Account ID or alias (Don't have?)

283067151160

☐ Remember this account

IAM username

admin-user

Password

Meri\_dad1

☒ Show Password [Having trouble?](#)

Sign in

Sign in using root user email

[Create a new AWS account](#)

By continuing, you agree to [AWS Customer Agreement](#) or other agreement for AWS services, and the [Privacy Notice](#). This site uses essential cookies. See our [Cookie Notice](#) for more information.

### Amazon Lightsail

Lightsail is the easiest way to get started on AWS

Learn more »

### Specify user details

#### User details

User name

admin-user

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ \_ - (hyphen)

☒ Provide user access to the AWS Management Console - optional  
If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

#### Console password

☐ Autogenerated password  
You can view the password after you create the user.

☒ Custom password  
Enter a custom password for the user.

\*\*\*\*\*

☐ Show password

☐ Users must create a new password at next sign-in - Recommended  
Users automatically get the [IAMUserChangePassword](#) policy to allow them to change their own password.

① If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. [Learn more](#)

Cancel Next

Europe (Stockholm) ▼

Account ID: 2830-6715-1160 ▼

admin-user

[Reset to default layout](#) [+ Add widgets](#)

### Applications (0) Info

Region: Europe (Stockholm)

Select Region

eu-north-1 (Current Region) ▼

[Find applications](#)

< 1 >

Name	Description	Region	Originati
------	-------------	--------	-----------



# CREATION DE L'INSTANCE EC2

## creation de l instance

Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name

my-app

Add additional tags

Application and OS Images (Amazon Machine Image)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

Search our full catalog including 1000s of application and OS images

Quick Start

Summary

Number of instances

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.9.2...[read more](#)

ami-0c7d68785ec07306c

Virtual server type (instance type)

t3.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Cancel

Launch instance

Edit inbound rules

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-07e4e64ebc9c8eda0	HTTP	TCP	80	Custom	
sgr-05e841f4c091b8e1c	SSH	TCP	22	Custom	

Add rule

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

hell@widad MINGW64 ~/Desktop/s5/cloud

\$ ssh -i "app-key.pem" ec2-user@13.51.235.243

Amazon Linux 2023

https://aws.amazon.com/linux/amazon-linux-2023

[ec2-user@ip-172-31-31-59 ~]\$

## configuration docker

```
[ec2-user@ip-172-31-31-59 ~]$ sudo yum install docker -y
Last metadata expiration check: 0:00:45 ago on Tue Nov 18 11:08:51 2025.
Dependencies resolved.
=====
Package                Arch      Version                               Repository    Size
=====
Installing:
docker                  x86_64    25.0.13-1.amzn2023.0.2              amazonlinux   46 M
Installing dependencies:

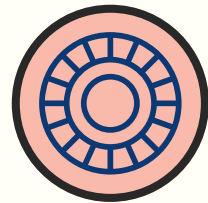
```

```
[ec2-user@ip-172-31-31-59 ~]$ sudo systemctl start docker
[ec2-user@ip-172-31-31-59 ~]$ sudo systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-31-59 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-31-59 ~]$
```

```
[ec2-user@ip-172-31-31-59 ~]$ docker --version
Docker version 25.0.13, build 0bab007
[ec2-user@ip-172-31-31-59 ~]$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
[ec2-user@ip-172-31-31-59 ~]$
```

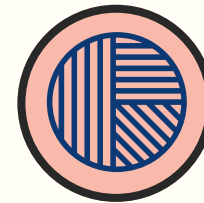
15

# CHOIX TECHNIQUES AWS



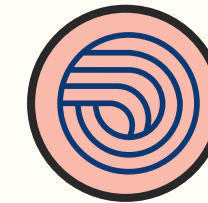
## **Instance : t3.micro**

Volume SSD, chiffré  
automatiquement.



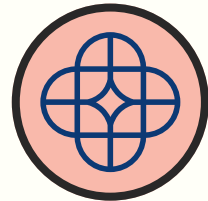
## **AMI : Amazon Linux 2023**

Stable, optimisé AWS,  
support long terme.



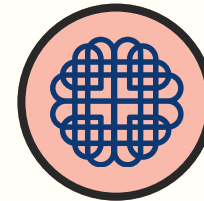
## **Région : eu-north-1 (Stockholm)**

Performances correctes,  
disponibilité élevée.



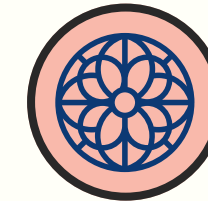
## **Stockage : 16 GiB gp3**

Ingoude Company  
Marketing Plan  
Implementation  
2024-2030



## **Réseau : VPC & Subnet par défaut**

IP publique assignée  
automatiquement.



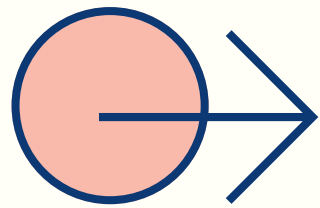
## **Connexion : SSH avec clé .pem**

Utilisateur ec2-user,  
accès sécurisé.



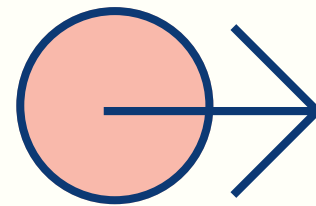
# DÉPLOIEMENT

# TRANSFERT ET CONSTRUCTION



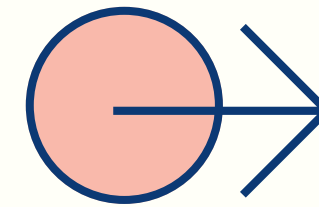
## Transfert sécurisé via SCP

Copie de tous les fichiers du projet (code, Dockerfile, données, embeddings) vers l'EC2 avec la clé PEM.



## Construction de l'image Docker

**DOCKER BUILD -T MY\_APP .**  
Installation des dépendances, configuration de l'environnement.



## Test de lancement du conteneur

Vérification du fonctionnement correct et isolation réseau validée.





# **CONFIGURATION**

## **OLLAMA ET NGROK**

# OLLAMA LOCAL

- Configuré Ollama pour accepter les connexions externes
- Ouvert le firewall Windows spécifiquement pour le port 11434
- Vérifié que le serveur était bien accessible réseau
- Lancé les modèles IA en mode serveur permanent

```
PS C:\Users\dell> [Environment]::SetEnvironmentVariable("OLLAMA_HOST", "0.0.0.0", "Machine")
PS C:\Users\dell>
```

```
PS C:\Users\dell> netstat -an | findstr :11434
TCP        0.0.0.0:11434          0.0.0.0:0              LISTENING
TCP        127.0.0.1:11434        0.0.0.0:0              LISTENING
TCP        [::]:11434            [::]:0                 LISTENING
```

```
PS C:\Users\dell> New-NetFirewallRule -DisplayName "Ollama" -Direction Inbound -Protocol TCP -LocalPort 11434 -Action Allow
```

```
PS C:\Users\dell> ollama serve
time=2025-11-21T11:40:56.041+01:00 level=INFO source=routes.go:1544 msg="server config" env="map[CUDA_VISIBLE_DEVICES: G
GML_VK_VISIBLE_DEVICES: GPU_DEVICE_ORDINAL: HIP_VISIBLE_DEVICES: HSA_OVERRIDE_GFX_VERSION: HTTPS_PROXY: HTTP_PROXY: NO_P
ROXY: OLLAMA_CONTEXT_LENGTH:4096 OLLAMA_DEBUG:INFO OLLAMA_FLASH_ATTENTION:false OLLAMA_GPU_OVERHEAD:0 OLLAMA_HOST:http:/
```

# NGROK

- Installé et configuré le client Ngrok
- Créé un compte Ngrok sécurisé avec authentification double
- Établi un tunnel chiffré vers notre serveur Ollama local
- Obtenu une URL publique sécurisée pour l'accès distant

```
ngrok - tunnel local ports to public URLs and inspect traffic

USAGE:
  ngrok [command] [flags]

COMMANDS:
  api          CLI to api.ngrok.com
  completion  generates shell completion code for bash or zsh
```

## Your Authtoken

Use this personal Authtoken to authenticate ngrok agents, SDKs, and the Kubernetes Operator for your own projects. Keep it secret, like a password.

35kXBdSkH6Hp3J4BDpn7DQzdtSk\_7cMfviWcL9vrF5Get22TB

Copy

```
PS C:\Users\dell> ngrok authtoken 35kXBdSkH6Hp3J4BDpn7DQzdtSk_7cMfviWcL9vrF5Get22TB
Authtoken saved to configuration file: C:\Users\dell\AppData\Local\ngrok\ngrok.yml
```

```
PS C:\Users\dell> ngrok http 11434
```

```
ngrok
♦ Put your secrets in vaults and (re)use them to transform traffic: https://ngrok.com/r/secrets

Session Status      online
Account             testliakolchi@gmail.com (Plan: Free)
Update              update available (version 3.33.0, Ctrl-U to update)
Version             3.24.0-msix
Region              Europe (eu)
Web Interface        http://127.0.0.1:4040
Forwarding           https://raisiny-unfoldable-margot.ngrok-free.dev -> http://localhost:11434
```

# ASSEMBLAGE CLOUD-LOCAL

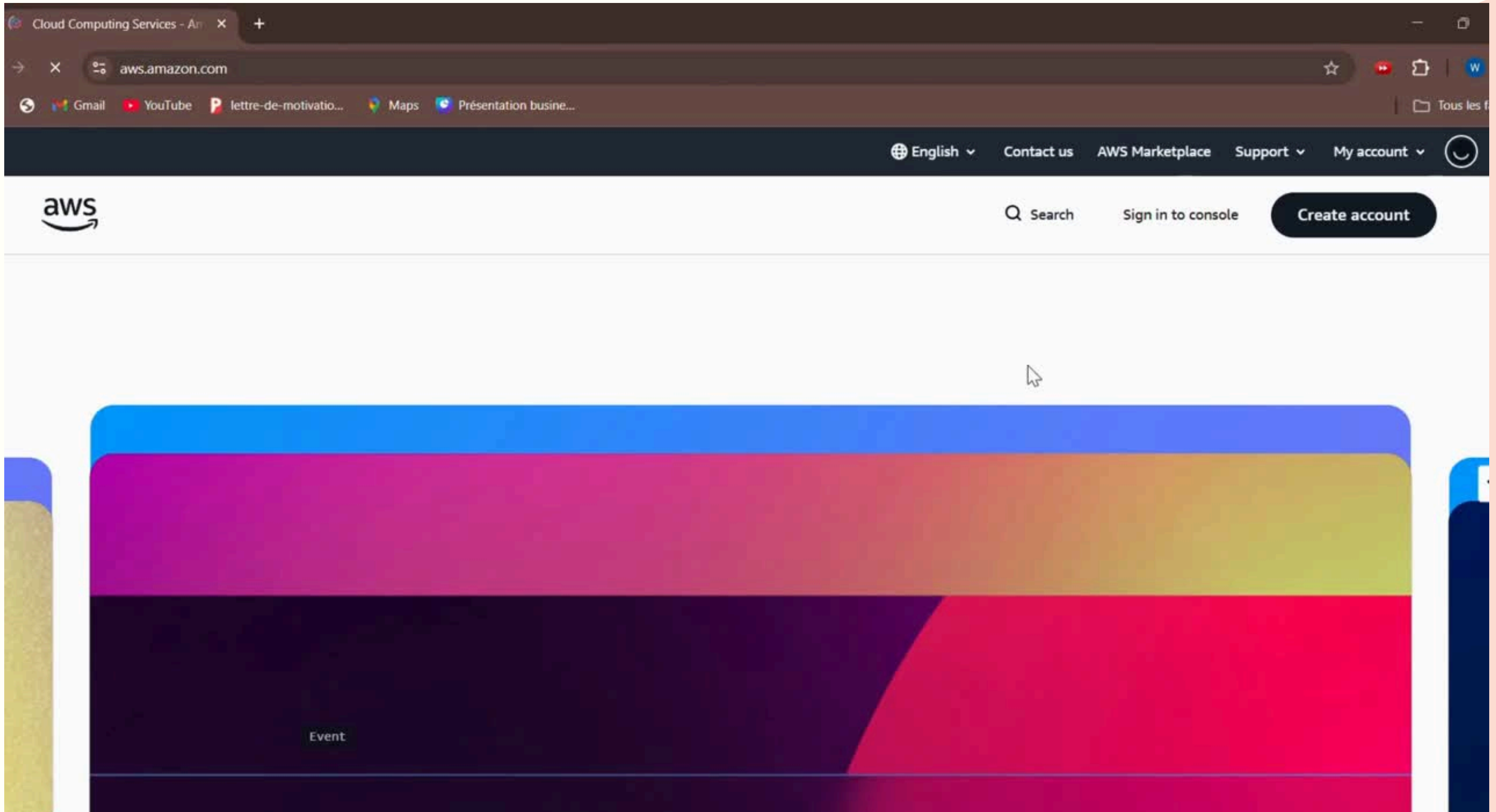
```
[ec2-user@ip-172-31-31-59 ~]$ docker run -d --name my_app -p 8501:8501 -e BASE_URL=https://raisiny-unfoldable-margot.ngrok-free.dev my_app  
5bca77425c3233c872133a4ecf7242f405048cfef75f7fc3143805d3287b693c  
[ec2-user@ip-172-31-31-59 ~]$
```

- Connecté l'application AWS EC2 à notre backend IA local
- Injecté l'URL Ngrok dans le conteneur Docker via variable d'environnement
- Lancé l'application complète avec une seule commande
- Vérifié que la communication cloud-local fonctionnait parfaitement



# **RÉSULTATS ET DÉMONSTRATION**







# **DÉFIS RENCONTRÉS ET SOLUTIONS**

# PROBLÈMES DE MÉMOIRE

## ***LIMITATION DES RESSOURCES AWS***

- L'instance t3.micro (1 Go RAM) ne suffisait pas pour exécuter Ollama et Streamlit simultanément.
- Crashes systématiques avec les modèles LLM.



***SOLUTION***

## ***LIMITATION DES RESSOURCES AWS***

## **PROBLÈMES DE MÉMOIRE**

### ***ARCHITECTURE HYBRIDE***

- **Backend IA local** : Ollama exécuté sur une machine personnelle plus puissante.
- **Frontend léger** : Streamlit sur EC2 dédié à l'interface utilisateur.

# PROBLÈMES DE MÉMOIRE

## ***LIMITES RÉSEAU ET CONTOURNEMENT***

- Restrictions réseau (NAT FAI, firewall, IP dynamiques) empêchaient la connexion directe EC2 ↔ machine locale.



***SOLUTION***

## ***LIMITES RÉSEAU ET CONTOURNEMENT***

# **PROBLÈMES DE MÉMOIRE**

## ***NGROK***

Tunnel HTTPS sécurisé

URL persistante malgré IP dynamiques

Chiffrement SSL pour protéger les données

Monitoring intégré pour suivre le trafic

- Déploiement d'une application IA avancée avec un budget minimal
- Combinaison réussie d'un frontend cloud AWS et d'un backend IA local
- Utilisation de Ngrok pour surmonter les contraintes matérielles
- Maintien de hautes performances malgré l'approche low-cost
- Preuve qu'une IA performante peut être déployée à moindre coût
- Solution pouvant servir de référence pour d'autres projets similaires

# CONCLUSION





**MERCI POUR VOTRE ATTENTION!**