Déploiement d'une application web NLP météo avec Kubernetes sur Google Cloud

- IKHOUANE Sansa
- KHELIFI Eya

Contexte et objectifs

Objectif principal:

- Déployer une application NLP météo complète (backend + frontend + bdd) sur un cluster Kubernetes

Objectifs techniques

- Conteneuriser une API NLP avec Docker
- Déployer l’application sur Google Kubernetes Engine (GKE)
- Mettre en place une architecture avec :
  - Deployments
  - Services
  - Ingress
- Automatiser l’infrastructure avec Terraform
- Tester le fonctionnement via HTTP et réseau interne Kubernetes

Présentation du projet

Le projet consiste à développer et déployer une application NLP météo.

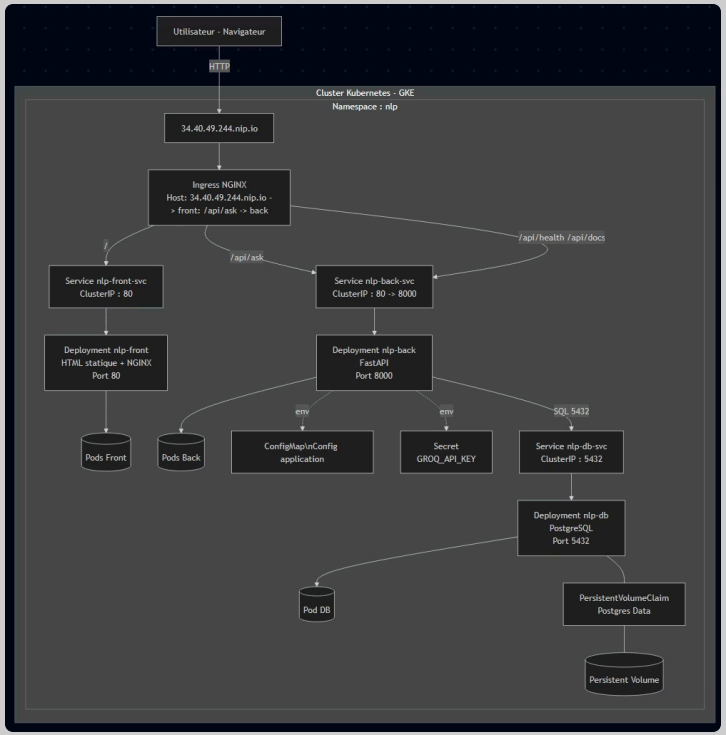
Elle permet à un utilisateur de :

- poser une question en langage naturel (ex : *“Quelle est la météo à Paris demain ?”*)
- recevoir une réponse structurée depuis une API NLP

L’application est composée de :

- un backend FastAPI qui traite les requêtes NLP
- un frontend web simple (HTML + JavaScript)
- une infrastructure Kubernetes pour l’hébergement

Architecture technique



Flux de requêtes

1) Accès utilisateur (frontend):

Exemple: L’utilisateur ouvre son navigateur et accède à l’application

Ce qu'il se passe:

La requête HTTP arrive sur l’Ingress NGINX et Il redirige la requête vers le service frontend

2) Envoi d’une question depuis le frontend:

Exemple: Quelle est la météo à Paris demain?”

Ce qu'il se passe:

Le frontend envoie une requête HTTP

Flux de requêtes

3) Passage par l’Ingress Kubernetes:

La requête arrive sur l’Ingress et l’Ingress détecte le préfixe /api.

Il route vers le backend.

4) Traitement par le backend NLP

L’API reçoit la requête.

Le backend :

- Analyse la question
- Vérifie qu’il s’agit d’une question météo
- Appelle les outils NLP / météo
- Génère une réponse

Flux de requêtes

5) Réponse Backend:

Après le traitement de la question NLP, le backend renvoie une réponse HTTP.

Cette réponse est générée par l'API FastAPI et contient la question originale et la réponse calculée par le backend NLP.

6) Chemin de retour de la réponse

La réponse suit le chemin inverse dans l'architecture :

backend → service kubernetes → ingress nginx → frontend

Le frontend affiche la réponse à l'utilisateur, l'utilisateur n'a aucune connaissance de Kubernetes.

Déploiement sur GKE

Rôle de GKE dans le projet:

- Hébergement du cluster Kubernetes
- Gestion automatique : des nœuds, des mises à jour et de la sécurité
- Haute disponibilité intégrée

1) Création du cluster GKE:

Le cluster Kubernetes est créé sur GKE à l'aide de Terraform.

Terraform permet de créer le cluster GKE, configurer la zone et le type de machines et activer les services Google nécessaires.

2) Application des manifestes:

Utilisation de kubectl apply -f pour déployer tous les objets kubernetes définis

3) Récupération de l'IP publique (Ingress)

L'Ingress crée automatiquement un LoadBalancer sur GCP afin d'accèder à l'application.

DEMO :)