Exercice Professionnel : Déploiement d'un Nginx Non-Root avec OpenShift et Terraform

🎯Objectif

Mettre en place une application web basée sur **Nginx non-root** dans l'environnement OpenShift Sandbox, en automatisant la création des ressources nécessaires avec **Terraform**. Ensuite, personnaliser la page d'accueil via un **ConfigMap**.

---

1. Pré-requis

- Un compte actif sur OpenShift Sandbox.
- `oc` CLI configuré (`oc login ...`).
- `terraform` installé et ajouté au `PATH` .
- Droits développeur dans le projet par défaut : `contact-walid-labidi-dev` .

---

2. Étapes du Déploiement (Version Avancée)

Étape 1 : Initialisation du projet Terraform

Dans PowerShell :

```
mkdir C:\terraform\nginx
cd C:\terraform\nginx
notepad main.tf
```

Étape 2 : Écriture du fichier `main.tf`

```
terraform {
  required_providers {
    kubernetes = {
      source  = "hashicorp/kubernetes"
      version = "2.27.0"
    }
  }
}

provider "kubernetes" {
  config_path = "~/.kube/config"
}

# ConfigMap pour page personnalisée
```

```
resource "kubernetes_config_map" "nginx_index" {
  metadata {
    name      = "nginx-index"
    namespace = "contact-walid-labidi-dev"
  }

  data = {
    "index.html" = <<EOT
<html>
  <head>
    <title>Bienvenue sur Nginx OpenShift</title>
  </head>
  <body style="font-family:sans-serif; text-align:center;">
    <h1 style="color:green;">Hello depuis OpenShift Sandbox 🚀</h1>
    <p>Déploiement avec Terraform et ConfigMap réussi.</p>
  </body>
</html>
EOT
  }
}

# Deployment (nginx non-root)
resource "kubernetes_deployment" "nginx_nonroot" {
  metadata {
    name      = "nginx-nonroot"
    namespace = "contact-walid-labidi-dev"
    labels = {
      app = "nginx-nonroot"
    }
  }

  spec {
    replicas = 1
    selector {
      match_labels = {
        app = "nginx-nonroot"
      }
    }

    template {
      metadata {
        labels = {
          app = "nginx-nonroot"
        }
      }

      spec {
        volume {
          name = "html-content"
          config_map {
            name = kubernetes_config_map.nginx_index.metadata[0].name
```

```
          }
        }

        container {
          name  = "nginx"
          image = "nginxinc/nginx-unprivileged:1.25-alpine"

          port {
            container_port = 8080
          }

          volume_mount {
            name       = "html-content"
            mount_path = "/usr/share/nginx/html/index.html"
            sub_path   = "index.html"
          }
        }
      }
    }
  }
}

# Service
resource "kubernetes_service" "nginx_svc" {
  metadata {
    name      = "nginx-nonroot"
    namespace = "contact-walid-labidi-dev"
  }

  spec {
    selector = {
      app = "nginx-nonroot"
    }

    port {
      port        = 80
      target_port = 8080
    }

    type = "ClusterIP"
  }
}

# Route OpenShift via local-exec
resource "null_resource" "nginx_route" {
  provisioner "local-exec" {
    command = "oc -n contact-walid-labidi-dev expose service/nginx-nonroot"
  }

  depends_on = [kubernetes_service.nginx_svc]
}
```

3. Exécution des Commandes

Initialiser Terraform

```
terraform init
```

Appliquer le plan

```
terraform apply -auto-approve
```

---

4. Vérification

Pods

```
oc get pods -n contact-walid-labidi-dev
```

Attendu : pod `nginx-nonroot` en état `Running`.

Service

```
oc get svc -n contact-walid-labidi-dev
```

Attendu : service `nginx-nonroot` exposant port 80.

Route

```
oc get route -n contact-walid-labidi-dev
```

Attendu : route publique `nginx-nonroot` générée.

---

5. Test de l'Application

Ouvrir dans le navigateur :

```
http://nginx-nonroot-contact-walid-labidi-dev.apps.sandbox-
m2.1530.p1.openshiftapps.com
```

La page **personnalisée** (texte vert et message Terraform) s'affiche au lieu de la page Nginx par défaut.

Résultat Final

- Déploiement automatisé avec Terraform.
- Page Nginx remplacée par une version personnalisée via **ConfigMap**.
- Route publique opérationnelle.

Points Clés

- **ConfigMap** → permet d'injecter du contenu statique (ex: `index.html`).
- **Volume + VolumeMount** → remplace le fichier par défaut de Nginx.
- **Sécurité** → image `nginx-unprivileged` tourne en non-root, compatible avec OpenShift Sandbox.
- **Automatisation complète** → Terraform gère tout, y compris la création automatique de la route via `oc expose`.