EXAMEN - Mise en place d'un pipeline CI/CD avec GitHub Actions

Contexte

Vous êtes DevOps Engineer dans une équipe qui développe une application web simple (HTML). Votre mission est de mettre en place un pipeline CI/CD avec GitHub Actions permettant :

1. La récupération du code source

2. La construction d'une image Docker

3. Le push de l'image vers Docker Hub

4. L'analyse de vulnérabilités avec Trivy

5. Le déploiement automatique sur un serveur via un self-hosted runner

Ressources fournies

- Téléchargez un exemple de site web à partir du lien https://html5up.net/
- Construire votre Dockerfile en vous inspirant de l'exemple ci-joint
- Créer un repository GitHub et mettez-y votre code source. Créer votre ficher wordflow.
- Créez un compte Docker Hub personnel
- Utilisez un github hosted runner(ubuntu-latest) pour le job de Build et un self-hosted runner pour le job de deploy.
- Un exemple complet est fournit sur le repo Github : https://github.com/moisawade/html5up-forty

Travail demandé

Partie 1 - CI : Build & Push (8 points)

Créer un workflow GitHub Actions déclenché sur push vers la branche main. Le workflow devra :

- Utiliser ubuntu-latest

- Définir les variables IMAGE_NAME, IMAGE_TAG, DOCKER_REGISTRY

- Se connecter à Docker Hub via secrets

- Builder l'image Docker

- Pusher l'image vers Docker Hub

Partie 2 - Scan de sécurité (4 points)

Ajouter une étape utilisant Trivy pour scanner l'image Docker avec les critères suivants :

- Sévérité HIGH et CRITICAL

- exit-code: 1 pour bloquer le pipeline en cas de vulnérabilité

Exemple: Job scan vulnerabilities avec trivy.

```yaml
- name: Run Trivy vulnerability scanner
  uses: aquasecurity/trivy-action@0.20.0
  with:
    image-ref: moisawade/ci-html-demo:latest
    format: table
    severity: CRITICAL,HIGH
    exit-code: 1
    vuln-type: os,library
    ignore-unfixed: true
```

Partie 3 - CD : Déploiement sur self-hosted runner (6 points)

Créer un second job nommé 'deploy' :

- Utiliser runs-on: self-hosted

- Dépendre du job CI via 'needs'

- Pull la dernière image depuis Docker Hub

- Stopper le conteneur existant (si présent)

- Supprimer l'ancien conteneur

- Lancer un nouveau conteneur exposé sur le port 80

Questions théoriques (2 points)

1. Quelle est la différence entre ubuntu-latest et self-hosted ?

2. Pourquoi séparer CI et CD en deux jobs ?

3. Quel est le rôle du mot-clé 'needs' ?