CI/CD

Un rapport **détaillé** des actions réalisé est attendu. Un lien vers un dépôt GitHub devra y figurer.

Prérequis

- docker
- docker-compose

Questions

1. Créez un fichier `docker-compose.yml` et ajoutez-y un service `db` s'appuyant sur l'image Docker `postrges:latest`.

2. Créez une base de données `city_api` avec une table `city` contenant les colonnes suivantes :

   - `id`, un entier non signé non nul, clé primaire de la colonne ;
   - `department_code`, une chaîne de caractères non nulle ;
   - `insee_code`, une chaîne de caractères ;
   - `zip_code`, une chaîne de caractères ;
   - `name`, une chaîne de caractères non nulle ;
   - `lat`, un flottant non nul ;
   - `lon`, un flottant non nul.

3. Dans le langage de votre choix, créez un service web ayant les spécifications suivantes :

   - `POST /city` avec pour corps de la requête un JSON au format décrit plus bas doit retourner un code `201` et enregistrer la ville dans la base de données ;
   - `GET /city` doit retourner un code `200` avec la liste des villes au format JSON ;
   - `GET /_health` doit retourner un code `204`.

   Vous pouvez trouver un exemple de JSON ici.

   Faîtes en sorte que votre service soit configurable avec les variables d'environnement suivantes :

   - `CITY_API_ADDR`, qui correspond à l'adresse d'écoute de votre serveur HTTP (par défaut, `127.0.0.1`) ;
   - `CITY_API_PORT`, qui correspond au port d'écoute de votre serveur HTTP (par défaut, `2022`) ;
   - `CITY_API_DB_URL`, qui correspond à l'URL de connexion vers la base de données (le service doit planter si non specifié) ;
   - `CITY_API_DB_USER`, qui correspond au nom d'utilisateur utilisé pour se connecter à la base de données (le service doit planter si non specifié) ;
   - `CITY_API_DB_PWD`, qui correspond au mot de passe utilisé pour se connecter à l base de données (le service doit planter si non specifié).

4. Écrivez les tests suivants :

   - un test qui s'assure que l'insertion dans la base de données fonctionne correctement ;
   - un test qui s'assure que la récupération de la liste des villes fonctionne correctement ;
   - un test qui s'assure que l'endpoint de healthcheck fonctionne correctement.

5. Écrivez un fichier `Dockerfile` à la racine de votre projet. Testez que votre image Docker est correcte.
6. Écrivez un workflow GitHub Actions `ci` pour qu'un linter soit exécuté à chaque push.
7. Modifiez le workflow pour que les tests s'exécutent à chaque push.
8. Modifiez le workflow pour qu'un build de l'image Docker soit réalisé à chaque push.
9. Modifiez le workflow pour que l'image Docker soit push sur `ghcr.io` avec pour tag `city-api:latest`.
10. Écrivez un workflow GitHub Actions `release` qui, lorsqu'un tag au format `vX.X.X` soit poussé build et push l'image Docker avec un tag `city-api:X.X.X`.