

Workflow GitLab

Auteur: REICHHART Damien - BENAD Kilian

Date de création: 23/04/2025 **Dernière modification:** 23/04/2025

Sommaire

1. [Introduction](#)
2. [Branches](#)
3. [Conventions de Commit](#)
4. [Pipeline CI/CD](#)
5. [Revue de Code](#)

Introduction

Ce document décrit le workflow GitLab mis en place pour le projet. Il définit les règles de gestion des branches, les conventions de commit et le processus de revue de code.

Branches

Branches Principales

- **main**: Branche de production
- **dev**: Branche de développement

Branches de Fonctionnalités

Format: **feature/nom-fonctionnalite**

Exemple:

```
git checkout -b feature/gestion-conteneurs
```

Branches de Correction

Format: **hotfix/nom-correction**

Exemple:

```
git checkout -b hotfix/correction-bug-conteneur
```

Conventions de Commit

Chaque message de commit doit suivre le format suivant :

[Type] Description concise

Description détaillée (si nécessaire)

Types de commit :

- [FIX]: Correction de bug
- [FEATURE]: Ajout de fonctionnalité
- [DOC]: Documentation
- [RELEASE]: Version
- [HOTFIX]: Correction urgente
- [REFACTOR]: Refactorisation
- [TEST]: Tests

Exemple :

[FEATURE] Ajout de la gestion des conteneurs Docker

- Implémentation de la création de conteneurs
- Ajout des tests unitaires
- Mise à jour de la documentation

Pipeline CI/CD

Le pipeline GitLab CI/CD est configuré dans le fichier `.gitlab-ci.yml` et comprend les étapes suivantes :

1. Build

- Installation des dépendances
- Compilation des assets

2. Test

- Tests unitaires
- Tests d'intégration
- Analyse de code statique

3. SAST

- Analyse de sécurité
- Vérification des vulnérabilités

4. Clean

- Nettoyage des ressources temporaires

5. Deploy

- Déploiement en environnement de test

- Déploiement en production (manuel)

Revue de Code

1. Création d'une Merge Request (MR)
2. Assignment des reviewers
3. Revue du code
4. Résolution des commentaires
5. Approbation de la MR
6. Fusion dans la branche cible