DEVOPS-040 — GitLab CI/CD : industrialiser le cycle de vie logiciel

Programme de formation — Telemach Learning

Code	DEVOPS-040
Titre	GitLab CI/CD : industrialiser le cycle de vie logiciel
Durée	21 heures — 3 jours
Format	Intra-entreprise — Présentiel (distanciel possible sur demande)
Public	Développeurs, ingénieurs DevOps, administrateurs systèmes, architectes, chefs de projet techniques
Prérequis	Pratique de Linux en ligne de commande, bases de Git (commit, branche, fusion), notions de Docker (DEVOPS-001 recommandé)
Niveau	Intermédiaire
Financement	OPCO — Plan de Développement des Compétences (PDC)
Organisme	Telemach Learning — NDA 11 92 27843 92
Formateur	Alexandre BOIGUES

Objectifs pédagogiques

À l'issue de cette formation, le participant est capable de :

- Organiser le travail collaboratif dans GitLab : groupes, droits, issues, merge requests et branches protégées
- Écrire, structurer et optimiser un pipeline GitLab CI/CD : stages, jobs, règles, cache, artefacts, dépendances entre jobs
- Factoriser des pipelines réutilisables avec les includes, les templates et les composants CI/CD
- Construire, publier et déployer une application conteneurisée vers des environnements de recette et de production, avec approbation
- Intégrer les contrôles de sécurité à la chaîne de livraison et administrer les runners qui l'exécutent

© ALEXANDRE BOIGUES 2026 TOUS DROITS RÉSERVÉS — NDA 11 92 27843 92 — v1.0

Jour 1 — Collaborer avec GitLab et écrire ses premiers pipelines

Les ateliers s'appuient sur une même application, Télémarché (interface web, API et base de données PostgreSQL) : versionnée dans GitLab dès le premier module, elle est testée, empaquetée, sécurisée puis déployée, module après module, jusqu'à une chaîne de livraison complète.

Module 1 — GitLab, plateforme de collaboration (3h30)

- L'offre GitLab : éditions Community et Enterprise, gitlab.com ou instance auto-hébergée
- Groupes, sous-groupes et projets ; rôles et permissions
- Rappels Git appliqués à GitLab : branches, fusion, rebase, résolution de conflits
- Issues, tableaux, jalons ; lier un commit ou une merge request à une issue
- Merge requests : revue de code, discussions, approbations, branches protégées
- Choisir un modèle de branches : trunk-based, GitLab Flow
- Atelier : organiser le projet Télémarché, travail en équipe par merge requests avec revue croisée

Module 2 — Premiers pipelines GitLab CI/CD (3h30)

- Runners et executors : qui exécute les jobs, et où
- Le fichier `.gitlab-ci.yml` : stages, jobs, image, script, before_script
- Variables CI/CD : prédéfinies, de projet et de groupe, masquées et protégées
- Artefacts et cache : rôles, différences, durée de conservation
- Déclencher au bon moment : `rules`, pipelines de merge request, jobs manuels et planifiés
- Lire et diagnostiquer un pipeline en échec : journal du job, éditeur de pipeline, validation de la syntaxe
- Atelier Télémarché : compiler et tester l'API, publier les rapports de tests et la couverture dans la merge request

© ALEXANDRE BOIGUES 2026 TOUS DROITS RÉSERVÉS — NDA 11 92 27843 92 — v1.0

Jour 2 — Structurer, construire et déployer

Module 3 — Structurer et accélérer les pipelines (3h30)

- Dépendances entre jobs avec `needs` : pipelines en graphe plutôt qu'en étapes
- Jobs parallèles et matrices
- Factoriser : `extends`, ancres YAML, `!reference`, `include` local, de projet et distant
- Composants CI/CD et catalogue CI/CD : publier et versionner des briques de pipeline partagées
- Pipelines parent-enfant et pipelines multi-projets pour les monorepos et les applications réparties
- Optimiser la durée d'un pipeline : stratégie de cache, interruption des pipelines obsolètes
- Atelier Télémarché : découper le pipeline en composants réutilisables, réduire sa durée et la mesurer

Module 4 — Construire, publier et déployer (3h30)

- Construire une image de conteneur dans un pipeline : Docker-in-Docker ou build sans privilèges (Buildah)
- Registre de conteneurs et registre de paquets GitLab
- Versionner et publier : tags, releases, journal des modifications
- Environnements et déploiements : recette, préproduction, production
- Review apps : un environnement éphémère par merge request
- Environnements protégés, approbations de déploiement, retour arrière
- Déployer sur Kubernetes depuis GitLab
- Atelier Télémarché : review app par merge request, déploiement en recette puis en production après approbation

© ALEXANDRE BOIGUES 2026 TOUS DROITS RÉSERVÉS — NDA 11 92 27843 92 — v1.0

Jour 3 — Sécuriser la chaîne et exploiter la plateforme

Module 5 — Sécuriser la chaîne de livraison (3h30)

- Gérer les secrets : variables masquées et protégées, coffre-fort externe, ce qu'il ne faut jamais écrire dans un dépôt
- Analyse statique du code (SAST) et détection de secrets
- Analyse des dépendances et nomenclature logicielle (SBOM CycloneDX)
- Analyse de l'image de conteneur et de l'infrastructure as code
- Tests de sécurité dynamiques (DAST) avec OWASP ZAP
- Scanners intégrés à GitLab et outils open source : ce que chaque édition apporte
- Seuils bloquants : arrêter la livraison sur une vulnérabilité critique, sans bloquer l'équipe pour du bruit
- Atelier Télémarché : ajouter les contrôles de sécurité au pipeline, corriger une vulnérabilité détectée et la voir disparaître du rapport

Module 6 — Exploiter la plateforme GitLab CI/CD (3h30)

- Types de runners : d'instance, de groupe, de projet ; tags et répartition des jobs
- Executors Docker et Kubernetes ; mise à l'échelle automatique des runners
- Sécurité des runners : mode privilégié, isolation, jetons d'enregistrement
- Infrastructure as code dans GitLab : Terraform ou OpenTofu, avec état géré par GitLab et plan relu en merge request
- GitOps : déployer avec Argo CD ou l'agent GitLab pour Kubernetes, sans donner au pipeline l'accès au cluster
- Suivre la santé des pipelines : durées, taux d'échec, jobs instables
- Atelier final : installer et enregistrer un runner de projet, puis livrer Télémarché de bout en bout, du commit à la production

© ALEXANDRE BOIGUES 2026 TOUS DROITS RÉSERVÉS — NDA 11 92 27843 92 — v1.0

Méthodes pédagogiques, évaluation et informations pratiques

Méthodes pédagogiques

- Alternance d'apports courts et d'ateliers pratiques (environ 60 % du temps en pratique)
- Un environnement isolé par participant : instance GitLab, runner et cluster Kubernetes, déployés à la demande avant la session puis détruits
- Travaux pratiques progressifs : pipeline de départ à compléter, solution de référence, script de vérification du résultat attendu
- Cas réels tirés de la pratique du formateur : défauts invisibles en intégration continue révélés au premier déploiement réel, vulnérabilités critiques détectées puis corrigées
- Dépôt de travaux pratiques remis aux participants pour continuer à pratiquer après la formation

Modalités d'évaluation

- Questionnaire de positionnement avant la formation
- Script de vérification à l'issue de chaque atelier
- Quiz de fin de formation
- Mise en situation finale : livrer Télémarché de bout en bout avec un pipeline complet et sécurisé
- Évaluation des acquis par le formateur en continu

Parcours

Cette formation fait suite à DEVOPS-001 — Conteneurisation avec Docker. Elle complète DEVOPS-011 — Intégrer des applications avec Kubernetes, qui situe Kubernetes dans une chaîne GitLab CI, et DEVOPS-010 — Orchestration avec Kubernetes, pour les profils qui industrialisent les déploiements sur un cluster.

Accessibilité

Pour toute situation de handicap, contacter Telemach Learning avant l'inscription afin d'étudier les adaptations possibles.

Financement

Formation éligible au Plan de Développement des Compétences (PDC).
OPCOs ciblés : OPCO 2i, Atlas, Constructys, Afdas, AKTO.

Telemach Learning — 24 Rue Chanzy, 92600 Asnières-sur-Seine
SIRET 522 540 061 00010 — NDA 11 92 27843 92
contact@telemach-learning.fr — www.telemach-learning.fr

© ALEXANDRE BOIGUES 2026 TOUS DROITS RÉSERVÉS — NDA 11 92 27843 92 — v1.0