DEVOPS & DÉPLOIEMENT

Git, GitHub Actions et Cyber Resilience Act : pipeline sécurisé pour développement .NET

Structurez votre travail collaboratif Git, industrialisez votre chaîne de build avec GitHub Actions et intégrez les exigences du Cyber Resilience Act dans votre cycle de développement. Formation pratique sur stack .NET, C# et WPF.

DURÉE	TARIF HT	NIVEAU	LANGUE	GROUPE	FORMAT
5j (35h)	**6000.00 €**	**Intermédiaire**	**Français**	**2-6**	**Formation**

1 PUBLIC VISÉ

- Développeurs et automaticiens travaillant sur applications desktop .NET, C#, VB.NET, WPF
- Équipes techniques produisant des logiciels embarqués ou superviseurs industriels
- Ingénieurs logiciel en charge de la chaîne de build et du déploiement
- Responsables techniques préparant la conformité au règlement CRA (UE) 2024/2847
- Toute équipe utilisant Git de manière basique et souhaitant structurer ses pratiques collaboratives

2 PRÉREQUIS

- Pratique régulière du développement logiciel, en particulier sur la stack .NET
- Connaissance basique de Git (clone, commit, push, pull) souhaitée mais non bloquante
- Aucune expérience préalable de GitHub Actions ni de DevSecOps requise
- Aucune connaissance préalable du Cyber Resilience Act requise
- Environnement de travail : poste avec Visual Studio 2022, accès Internet et compte GitHub

3 OBJECTIFS PÉDAGOGIQUES

- Maîtriser le travail collaboratif avec Git : branches, pull requests, résolution de conflits
- Mettre en place une stratégie de branches adaptée à un contexte de logiciel desktop industriel
- Industrialiser le cycle de build, test et release avec GitHub Actions
- Configurer les outils de sécurité applicative : CodeQL, Dependabot, secret scanning
- Produire et exploiter un SBOM CycloneDX pour un projet .NET
- Comprendre les obligations du Cyber Resilience Act et leurs implications techniques
- Identifier les exigences CRA applicables à un produit logiciel desktop
- Préparer le processus de signalement des vulnérabilités prévu à l'article 14 du CRA

4 PROGRAMME DÉTAILLÉ

Jour 1 (7h) – Git collaboratif : reconstruire des fondations solides

Module 1 – Modèle mental et fondamentaux Git (3h30)

Comprendre Git en profondeur (1h)

- Pourquoi Git n'est pas un outil de sauvegarde mais un graphe de snapshots
- Les objets fondamentaux : blob, tree, commit et leurs relations
- Le pointeur HEAD et le mécanisme des références
- Les trois zones : répertoire de travail, index, dépôt local

Manipulation au quotidien (1h30)

- Cycle de vie d'un fichier : untracked, modified, staged, committed
- Commandes essentielles : status, add, restore, commit, log, diff
- Conventions de messages de commit : standard Conventional Commits
- Gestion du fichier .gitignore pour projets Visual Studio (.vs, bin, obj, packages)
- Atelier pratique : initialiser un dépôt, premiers commits, navigation dans l'historique

Récupération en cas d'erreur (1h)

- Annuler une modification non commitée
- Modifier le dernier commit avec amend
- Mettre de côté son travail avec stash
- Retrouver des commits perdus avec reflog

Module 2 – Stratégies de branches et résolution de conflits (3h30)

Travailler avec les branches (1h)

- Création, navigation et suppression de branches locales
- Comparaison des stratégies : Git Flow, GitHub Flow, Trunk-Based
- Recommandation pour un contexte logiciel desktop industriel
- Conventions de nommage des branches en équipe

Merge et rebase (1h)

- Mécanisme du merge : commit de fusion et fast-forward
- Option --no-ff et préservation de la traçabilité des features
- Le rebase : réécriture d'historique et linéarisation
- Règle d'or : ne jamais rebaser une branche partagée
- Quand utiliser merge, quand utiliser rebase

Résolution de conflits en pratique (1h30)

Comprendre pourquoi un conflit apparaît

- Lecture et interprétation des marqueurs de conflit
- Résolution avec l'outil intégré à Visual Studio 2022
- Atelier en binôme : créer volontairement un conflit puis le résoudre proprement
- Cas concrets de conflits récurrents et comment les éviter

Jour 2 (7h) – GitHub : plateforme et collaboration

Module 3 – Organisation GitHub et gouvernance (2h)

Migration des comptes vers une organisation (1h)

- Comptes individuels GitHub Free : les limites pour le travail professionnel
- Création d'une organisation GitHub Team : structure, membres, équipes
- Risques juridiques du stockage de code professionnel sur comptes personnels
- Atelier guidé : création d'une organisation et transfert d'un dépôt

Authentification sécurisée (1h)

- Limites de l'authentification par mot de passe
- Personal Access Token : génération, portée et durée
- Authentification SSH avec clé ed25519
- Git Credential Manager pour Windows
- Configuration de Visual Studio 2022 avec authentification moderne

Module 4 – Pull requests et code review (3h30)

Anatomie d'une pull request (1h)

- Cycle de vie complet d'une PR : ouverture, revue, itération, merge
- Template de pull request en .github/pull_request_template.md
- Stratégies de merge : create a merge commit, squash, rebase
- Recommandation pour préserver la traçabilité des features

Code review structurée (1h30)

- Les sept dimensions d'une revue : fonctionnel, architectural, sécurité, performance, lisibilité, tests, style
- Conduite constructive : critiquer le code et non la personne
- Distinction des niveaux d'importance : blocker, suggestion, nit, question
- Utilisation des suggestions de modifications applicables en un clic
- Atelier en binôme : revue croisée d'une PR contenant des défauts plantés volontairement

Protection des branches (1h)

- Configuration des branch protection rules sur main et develop

Exigences d'approbation, de checks CI, de résolution des conversations

- Le fichier CODEOWNERS : automatisation des reviewers requis
- Configuration pratique sur leur organisation GitHub

Module 5 – Issues et gestion de projet (1h30)

Gestion des issues (45min)

- Création d'issues : titre, description, labels, milestones
- Liaison automatique entre PR et issues (Closes #123)
- Système de labels pour un projet industriel : type, priorité, module

Conventions de commits en équipe (45min)

- Format Conventional Commits : type(scope): description
- Rédaction d'une charte Git d'équipe
- Intérêt opérationnel : génération automatique de changelog, versioning sémantique

Jour 3 (7h) – Tests automatisés et GitHub Actions

Module 6 – Sensibilisation aux tests automatisés (2h30)

Pourquoi automatiser les tests (1h)

- Limites des tests manuels en équipe de développement
- Régression, rapidité, confiance : ce que les tests automatisés apportent
- Pyramide des tests : unitaires, intégration, end-to-end
- Notion de code testable et de conception orientée tests

Premiers tests xUnit (1h30)

- Création d'un projet de tests dans une solution .NET
- Anatomie d'un test xUnit : Fact, Theory, assertions
- Exécution des tests depuis Visual Studio et en ligne de commande
- Mesure de couverture de code avec coverlet
- Atelier : écriture de premiers tests sur une classe métier de démo

Module 7 – Anatomie de GitHub Actions (1h30)

Concepts fondamentaux (45min)

- Workflow, event, job, step, action, runner : la hiérarchie
- Choix du runner : windows-latest pour WPF, ubuntu-latest pour analyses
- Coût comparé des runners Windows et Linux
- La marketplace des actions et la sécurité des actions tierces

Structure d'un workflow YAML (45min)

- Triggers : push, pull_request, schedule, workflow_dispatch
- Définition des jobs et steps
- Variables d'environnement et contextes prédéfinis
- Lecture commentée d'un workflow CI minimal

Module 8 – Pipeline CI pour solution .NET (3h)

Construction d'un workflow de build (1h30)

- Step de checkout du code
- Configuration du SDK .NET avec actions/setup-dotnet
- Cache des packages NuGet pour accélérer les builds
- Restore, build et test d'une solution .NET en ligne de commande
- Publication des résultats de tests dans l'interface GitHub

Gestion des artefacts (45min)

- Production des binaires en sortie de pipeline
- Upload d'artefacts avec actions/upload-artifact
- Durée de rétention et bonnes pratiques

Atelier guidé sur leur stack (45min)

- Mise en place d'un workflow CI sur un projet .NET de démonstration
- Vérification de l'exécution et lecture des logs
- Diagnostic d'un échec de build courant

Jour 4 (7h) – Sécurité du pipeline et Cyber Resilience Act

Module 9 – Sécurité technique du pipeline (3h30)

Modèle de menaces de la chaîne logicielle (1h)

- Périmètre de la supply chain d'un logiciel desktop .NET
- Catégories de menaces : vulnérabilités du code, dépendances vulnérables, dépendances malveillantes
- Compromission du pipeline, falsification des artefacts, mises à jour piégées
- Cas concrets : Log4Shell, SolarWinds, compromissions npm et PyPI

SCA avec Dependabot pour NuGet (1h)

- Activation des alertes Dependabot sur un dépôt
- Configuration du fichier .github/dependabot.yml pour package-ecosystem nuget
- Lecture et qualification d'une alerte de vulnérabilité
- Audit complémentaire avec dotnet list package --vulnerable --include-transitive
- Atelier : configuration de Dependabot sur un dépôt de démonstration

SAST avec CodeQL et secret scanning (1h30)

- Présentation de CodeQL et de son fonctionnement
- Configuration d'un workflow CodeQL pour languages csharp
- Suites de requêtes : default, security-extended, security-and-quality
- Exemples de vulnérabilités détectées en C# : injection SQL, path traversal, désérialisation
- Activation du secret scanning et de la push protection
- Démonstration : tentative de push d'un secret bloquée par GitHub

Module 10 – SBOM et signature de code (1h30)

Génération de SBOM pour projet .NET (1h)

- Qu'est-ce qu'un SBOM et pourquoi il est exigé par le CRA
- Formats normalisés : CycloneDX (OWASP) et SPDX (Linux Foundation)
- Génération avec CycloneDX .NET : dotnet CycloneDX sur une solution
- Lecture du SBOM produit : dépendances directes vs transitives
- Intégration dans le pipeline et archivage pour conformité

Signature Authenticode et provenance (30min)

- Pourquoi signer : authenticité, intégrité, SmartScreen
- Types de certificats : OV, EV, signature cloud Azure Trusted Signing
- Démonstration : signature d'un exécutable avec signtool
- Attestation de provenance SLSA avec actions/attest-build-provenance

Module 11 – Cyber Resilience Act appliqué (2h)

Présentation du règlement (45min)

- Règlement (UE) 2024/2847 : entrée en vigueur et calendrier complet
- Périmètre : produits comportant des éléments numériques mis sur le marché UE
- Échéances clés : 11 septembre 2026 (signalement) et 11 décembre 2027 (conformité)
- Classification des produits : par défaut, important classe I et II, critique
- Sanctions : jusqu'à 15 M€ ou 2,5% du CA mondial

Obligations des fabricants (45min)

- Annexe I Partie I : 13 exigences essentielles relatives au produit
- Annexe I Partie II : 8 exigences de gestion des vulnérabilités
- Documentation technique Annexe VII et conservation 10 ans
- Mécanisme de notification 24h / 72h / 14 jours via la plateforme ENISA
- Politique de divulgation coordonnée et fichier security.txt

Atelier de simulation (30min)

Rédaction d'une notification initiale 24h à partir d'une vulnérabilité fictive sur une dépendance NuGet

- Cartographie des outils techniques vus en formation et des exigences CRA satisfaites
- Discussion ouverte sur le cas du logiciel desktop industriel

Jour 5 (7h) – Application sur projets réels et plan d'action

Module 12 – Mise en place du pipeline sur un projet client (3h30)

Atelier intégrateur (3h30)

- Sélection d'un projet .NET réel de l'équipe pour servir de cas d'application
- Création de l'organisation GitHub Team et migration du dépôt
- Configuration des branch protection rules et CODEOWNERS adaptés à l'équipe
- Mise en place du workflow CI avec build, tests et publication d'artefacts
- Activation de CodeQL et Dependabot pour les NuGet
- Génération du premier SBOM CycloneDX et archivage

Module 13 – Charte d'équipe et plan d'action (3h30)

Rédaction de la charte Git d'équipe (1h30)

- Stratégie de branches retenue et conventions de nommage
- Format des messages de commit et règles de pull request
- Modalités de code review et délais de validation
- Production d'un document partagé adopté par l'équipe

Plan d'action CRA personnalisé (1h30)

- Cartographie des produits de l'équipe et classification CRA probable
- Identification des écarts entre pratiques actuelles et exigences CRA
- Feuille de route sur 18 mois jusqu'à l'échéance de décembre 2027
- Définition des rôles internes : référent sécurité, référent vulnérabilités

Bilan et perspectives (30min)

- Synthèse des acquis et points à approfondir individuellement
- Ressources pour poursuivre la montée en compétences
- Évaluation à chaud et tour de table des participants

5 COMPÉTENCES VISÉES

- Travailler en équipe sur un dépôt Git partagé sans générer de conflits récurrents
- Conduire une code review structurée via les pull requests GitHub
- Concevoir et maintenir un workflow GitHub Actions pour une solution .NET
- Intégrer les contrôles de sécurité automatisés dans un pipeline CI/CD
- Générer automatiquement un SBOM à chaque release et l'archiver pour conformité
- Rédiger une notification de vulnérabilité conforme aux délais 24h/72h/14j du CRA
- Cartographier les obligations CRA applicables à un produit logiciel d'entreprise

6 MODALITÉS PÉDAGOGIQUES

Formation délivrée en présentiel intra-entreprise sur le site du client. Le formateur alterne entre méthode démonstrative (manipulation Git, GitHub et outils de sécurité en direct, projetée et commentée), méthode interrogative (questionnement actif des stagiaires) et méthode active (chaque participant manipule sur son propre poste tout au long de la formation). L'accent est mis sur la pratique sur la stack technologique réelle des stagiaires (.NET, C#, VB.NET, WPF, Visual Studio 2022) et sur l'application immédiate des concepts à leurs projets internes.

7 MOYENS PÉDAGOGIQUES ET TECHNIQUES

- Support de cours numérique complet remis aux participants au format PDF
- Fiche de référence des commandes Git essentielles
- Modèles de workflows GitHub Actions prêts à l'emploi pour projets .NET
- Checklist de conformité au Cyber Resilience Act
- Glossaire technique et réglementaire
- Dépôt GitHub de démonstration avec projet .NET et vulnérabilités plantées pour les ateliers
- Templates : pull request, CODEOWNERS, dependabot.yml, security.txt
- Modèle de notification de vulnérabilité conforme à l'article 14 du CRA
- Accès à la plateforme pédagogique LaPolaris (supports, ressources, émargement)

8 MODALITÉS D'ÉVALUATION

- En cours de formation : ateliers pratiques à chaque module avec correction collective
- Évaluations formatives par manipulation directe des outils sur les postes des participants
- Quiz de validation des connaissances en fin de chaque journée
- Atelier intégrateur du jour 5 : mise en place complète d'un pipeline CI sur un projet réel de l'équipe
- Production d'une charte Git d'équipe et d'un plan d'action CRA personnalisé en fin de parcours
- Questionnaire d'évaluation à chaud en clôture de formation

9 CRITÈRES D'ÉVALUATION

- Capacité à conduire un travail collaboratif Git sans générer de conflits récurrents
- Aptitude à concevoir et lire un workflow GitHub Actions adapté à une solution .NET
- Maîtrise des outils de sécurité automatisée : Dependabot, CodeQL, secret scanning
- Capacité à générer et exploiter un SBOM CycloneDX sur un projet .NET
- Compréhension claire du périmètre, des échéances et des obligations du Cyber Resilience Act
- Aptitude à rédiger une notification de vulnérabilité conforme aux exigences de l'article 14
- Production en fin de formation d'une charte d'équipe et d'un plan d'action CRA opérationnels

10 MODALITÉS DE VALIDATION

Attestation de fin de formation délivrée à l'issue du parcours, conditionnée à une assiduité d'au moins 80 % et à la réalisation de l'atelier intégrateur du jour 5. L'attestation précise les objectifs atteints et les compétences acquises par chaque participant. Un document de synthèse personnalisé est également remis : charte Git d'équipe et plan d'action CRA spécifique à l'organisation cliente.

11 SUIVI ET ACCOMPAGNEMENT

- Feuilles d'émargement signées par demi-journée pour les sessions en présentiel
- Traçabilité des activités pédagogiques réalisées et des ateliers conduits
- Attestation d'assiduité délivrée en fin de formation
- Suivi individuel par la vérification des dépôts GitHub configurés par chaque participant
- Échange en fin de chaque journée pour ajustement du contenu du jour suivant

12 CONDITIONS D'ACCÈS

Formation accessible sur inscription directe, sans prérequis administratif particulier. Le financement peut être pris en charge par l'employeur dans le cadre d'un plan de développement des compétences, ou en autofinancement avec possibilité de paiement en plusieurs fois.

13 DÉLAIS D'ACCÈS

Inscription possible jusqu'à **15 jours ouvrés** avant le début de la session pour permettre la préparation logistique et l'adaptation du contenu au contexte client. Pour toute demande spécifique, nous contacter directement.

ACCESSIBILITÉ · HANDICAP

Nos formations sont accessibles aux personnes en situation de handicap. Pour toute situation nécessitant un aménagement (matériel, temporel ou pédagogique), nous vous invitons à nous contacter avant l'inscription afin d'étudier les adaptations possibles. Référent handicap : contact@lapolaris.fr

LaPolaris
TÉL. +33762584798
EMAIL contact@lapolaris.fr
WEB lapolaris.fr