



Module: 324

Plan CI/CD

DevOps

Qu'est-ce qu'un plan CI/CD ?

Un plan CI/CD (Intégration Continue / Déploiement Continu) est une stratégie qui décrit les étapes, les outils et les processus nécessaires pour automatiser et optimiser le développement, les tests et le déploiement d'une application.

L'objectif est de permettre des mises à jour fréquentes, sûres et rapides du code en production, tout en assurant une haute qualité.

Plan Ci/CD

Quelles sont les étapes ?

Plan CI/CD

Définir les Objectifs et les Résultats

Identifiez les objectifs principaux, tels que la fréquence des déploiements, les besoins en rollback, la sécurité, et la performance attendue.

Déterminez également les contraintes, comme les technologies, les objectifs opérationnels.

Plan CI/CD

Analyser l'Architecture de l'Application

Technologie et Frameworks : Étudiez les frameworks spécifiques de chaque application (Flask pour Python, Spring pour Java, Node pour VueJS, etc.) et leurs implications sur le processus CI/CD.

Certaines technologies peuvent imposer des étapes de build plus longues ou des configurations d'environnement complexes.

Plan CI/CD

Analyser l'Architecture de l'Application

Dépendances : Identifiez les dépendances externes, comme les bases de données, API externes, micro-services, les bibliothèques et les configurations spécifiques.

Plan CI/CD

Analyser l'Architecture de l'Application

Type d'Application : Selon qu'il s'agisse d'une application web, d'une API, ou d'un service de backend, le workflow et les besoins CI/CD varieront.

Par exemple, un frontend Vue.js nécessitera un packaging et une optimisation de ressources différentes par rapport à un backend Flask.

Plan CI/CD

Analyser l'Architecture de l'Application

Automatisation des Tests : Définissez des tests unitaires, d'intégration et de régression dans chaque pipeline pour garantir que le code est fiable et prêt pour le déploiement.

Par exemple: PyTest pour Python, JUnit pour Java, Jest pour Node.js, ou Cypress pour les tests e2e.

Plan CI/CD

Évaluer les Contraintes Techniques et les Environnements

Infrastructure Cible : Décidez où les applications seront hébergées (on-premise, cloud, conteneurisé).

Ceci influencera les outils de déploiement et la manière de gérer les environnements:

- Linux ou Windows;
- Docker, Kubernetes pour les conteneurs,
- Azure, Google Cloud Service, AWS pour l'hébergement.

Plan CI/CD

Évaluer les Contraintes Techniques et les Environnements

Environnements de Déploiement : Définissez les environnements:

- Développement
- Test
- Pré-production (staging)
- Production

et assurez-vous qu'il existe une cohérence entre eux pour éviter les problèmes de configuration.

Plan CI/CD

Évaluer les Contraintes Techniques et les Environnements

Sécurité et Conformité : Intégrez des contrôles de sécurité :

- scan de vulnérabilités
- audit de dépendances
- Formatage du code (lint)

dans le workflow CI/CD pour répondre aux exigences de conformité.

Plan CI/CD

Construire le Workflow CI/CD

Pipeline de Build : Créez une première étape de pipeline qui assemble et package l'application.

Automatisation du Déploiement en Production : Envisagez un processus de validation manuel pour le déploiement en production.

Documentation des Processus : Documentez chaque étape du workflow, les outils, les scripts utilisés, et les procédures de validation pour permettre à l'équipe de bien comprendre le pipeline.