Formation **Applications Cloud-Native**

La formation "Applications Cloud-Native", d'une durée de deux jours, offre aux participants une compréhension approfondie des concepts du cloud-native et de ses avantages pour les entreprises modernes. Elle couvre des sujets clés tels que la conteneurisation et l'orchestration avec Docker et Kubernetes, ainsi que la conception d'architectures microservices et serverless pour répondre aux besoins d'évolutivité et de résilience. Les participants apprendront également à appliquer les Design Patterns du Cloud, à intégrer la sécurité via DevSecOps, et à planifier la migration d'applications existantes vers une architecture cloud-native. Des techniques de test de la résilience, comme le Chaos Engineering, sont également abordées pour assurer la tolérance aux pannes des systèmes en production.

Durée
2 jours

Objectifs pédagogiques

- Comprendre les concepts fondamentaux du cloud-native
- Maîtriser les technologies de conteneurisation et d'orchestration
- Concevoir et adapter des architectures cloud-native
- Appliquer les Design Patterns du Cloud et connaître les caractéristiques essentielles des applications cloud-native
- Mettre en oeuvre les principes des architectures microservices et des 12-factor apps
- Explorer le modèle Serverless et intégrer la sécurité via le DevSecOps

Public
Développeurs, chefs de projet, ingénieurs DevOps, admins

Prérequis
Bonnes connaissances en développement logiciel et des concepts liés au cloud, ainsi qu'une familiarité avec les environnements de conteneurisation comme Docker et Kubernetes.

Programme de formation

Phase d'inclusion

Accueil des participants, présentation des objectifs et contextes professionnels de chacun.

Fondamentaux du Cloud-Native

Définitions et concepts du cloud-native
Introduction aux principes du cloud-native.
Avantages et raisons pour adopter une approche cloud-native.
Conteneurs et orchestrations
Explication des conteneurs (Docker, Podman).
Outils d'orchestration (Kubernetes, Docker Swarm).
Gestion des conteneurs à grande échelle.
Les plateformes cloud-native
Services PaaS, CaaS et IaaS.
Plateformes populaires (Google Cloud, AWS, Azure).
Avantages des environnements cloud-native gérés.
Les architectures cloud-native
Types d'architectures (monolithique, microservices, serverless).
Adaptation de l'architecture selon les besoins métiers.
Exemple de transition d'une architecture traditionnelle vers une architecture cloud-native.

Conception et caractéristiques des applications cloud-native

Les Design Patterns du Cloud
Circuit Breaker, Retry Pattern, Sidecar Pattern, etc.
Mise en oeuvre des patterns pour une meilleure résilience et scalabilité.
Les caractéristiques des applications cloud-native
Scalabilité, tolérance aux pannes, et disponibilité.
Automatisation et gestion des ressources.

Approches et architectures modernes

Les architectures microservices
Décomposition d'applications en microservices indépendants.
Avantages et défis des microservices dans le Cloud.
Les 12-factor apps
Analyse des 12 principes pour construire des applications cloud prêtes à être déployées et évolutives.
Application pratique dans la conception d'une architecture.
Le Serverless
Présentation du modèle Serverless et des cas d'usage.
Différence entre Serverless et conteneurs.
Plateformes Serverless (AWS Lambda, Google Cloud Functions).

DevOps avancé, migration et déploiement

Le DevSecOps
Intégration de la sécurité dans les pratiques DevOps.
Outils et bonnes pratiques pour renforcer la sécurité des pipelines CI/CD.
Les stratégies de migration vers une architecture cloud-native
Méthodologies pour migrer une application existante vers le cloud-native.
Stratégies de découpage des monolithes en microservices.
Minimisation des risques pendant la migration.
Les modèles de déploiement
Introduction aux différents modèles de déploiement :
Blue-Green : bascule entre deux environnements pour minimiser les interruptions.
Canary : introduction progressive d'une nouvelle version sur une petite portion d'utilisateurs.
Rolling : remplacement progressif des versions sans interruption.
Shadow : test de la nouvelle version sans impact sur les utilisateurs réels.

Choix des modèles en fonction des besoins d'évolutivité et des risques.
Les principes du Chaos Engineering
Introduction au Chaos Engineering pour tester la résilience des systèmes.

Outils (Gremlin, Chaos Monkey) et cas pratiques.
Mise en place de tests de tolérance aux pannes pour garantir la stabilité des systèmes en production.

Moyens et méthodes pédagogiques

- La formation alterne entre présentations des concepts théoriques et mises en application à travers d'ateliers et exercices pratiques (hors formation de type séminaire).
- Les participants bénéficient des retours d'expérience terrains du formateur ou de la formatrice
- Un support de cours numérique est fourni aux stagiaires

Modalités d'évaluation

- **En amont de la session de formation**, un questionnaire d'auto-positionnement est remis aux participants, afin qu'ils situent leurs connaissances et compétences déjà acquises par rapport au thème de la formation.

- **En cours de formation**, l'évaluation se fait sous forme d'ateliers, exercices et travaux pratiques de validation, de retour d'observation et/ou de partage d'expérience, en cohérence avec les objectifs pédagogiques visés.

- **En fin de session**, le formateur évalue les compétences et connaissances acquises par les apprenants grâce à un questionnaire reprenant les mêmes éléments que l'auto-positionnement, permettant ainsi une analyse détaillée de leur progression.