Conception d'applications Cloud Native («CACN»)

Ce cours vous permettra de comprendre les différents concepts derrière le développement d'une application web distribuée. Les concepts de micro-services, serverless, object-storage, queue management, cache, edge computing et monitoring seront abordés.

Durée: 2 jours
Prix : 1'600.– excl. 8.1% TVA
Documents : Support de cours numérique Digicomp

Contenu

1. Conteneurs
    - Présentation des containers
    - Concepts principaux des containers
    - Création d'un container pour l'application
    - Publication du container dans un registry
    - Utilisation des containers dans l'application fictive
2. Micro-services
    - Présentation du paradigme micro-services
    - Découpage d'une application monolithique en micro-services
    - Notions de "API First"
    - Gestion des dépendances entre micro-services
    - Mise en place d'une API Gateway
    - Notions de messages et de callback entre les micro-services
    - Utilisation des micro-services dans l'application fictive
3. CDN et Edge computing
    - Présentation des différents types de CDN et des offres de Edge computing
    - Optimisation des performances de l'application web
    - Stratégie de caching HTTP
    - Savoir utiliser les bénéfices du Edge computing
    - Utilisation du cache HTTP dans l'application fictive
4. Applications sans état
    - Présentation du concept stateless
    - Conception d'une application sans état
    - Paradigmes usuels pour ne pas avoir d'état
    - Restauration de l'état basé sur la session
    - Utilisation du concept stateless dans l'application fictive
5. Applications multi-instances
    - Conception d'une application multi-instances (montée en charge horizontal)
    - Paradigmes usuels d'une application multi-instances
    - Synchronisation entre les différentes instances
    - Utilisation du concept multi-instances dans l'application fictive
6. Applications Serverless
    - Présentation du concept serverless
    - Présentation des différents frameworks serverless
    - Discussion sur les cas d'utilisation approprié pour faire du serverless
    - Paradigmes usuels du serverless
7. Stockage des fichiers
    - Présentation du concept d'object storage
    - Déploiement de MiniO
    - Utilisation de l'object storage dans l'application fictive
8. Gestion des queues de Job

   - Présentation du concept de jobs
   - Déploiement du gestionnaire de queue RabbitMQ
   - Utilisation du concept de jobs dans l'application fictive
9. Gestion du cache applicatif
   - Présentation du concept de caching
   - Discussions sur les meilleures approches pour faire du cache
   - Déploiement du serveur Redis
   - Utilisation du concept cache applicatif dans l'application fictive
10. Logging
   - Présentation du concept de remote logging
   - Déploiement de la stack ELK
   - Implémentation du concept de logging centralisé dans l'application factive
11. Monitoring
   - Présentation du monitoring distribué via des seuils
   - Comparaison avec les systèmes de monitoring standard
   - Déploiement de la stack Grafana/Prometheus/AlertManager
   - Implémentation du concept de monitoring dans l'application factive
12. Conclusion
   - Comparaison avec d'autres langages de programmation (Java, C#, PHP)
   - Discussion sur les avantages et inconvénients
   - Test des connaissances acquises

Objectifs

- Comprendre les différents outils à disposition
- Savoir utiliser les technologies présentées à bon escient
- Éviter les pièges communs lors du déploiement de ces technologies
- Pouvoir faire de la montée en charge à la demande sur une application web
- Savoir surveiller une application dans un contexte de reconfiguration automatique

Méthodologie & Didactique

Le cours est composé d'une partie théorique puis sur la création guidée d'une application web fictive intégrant toutes ces technologies. Un container Docker sera créé contenant l'application et docker-compose sera utilisé pour illustrer les dépendances.
Le langage de programmation retenu est Python et tous les concepts qui sont vus durant ce cours sont mis en relation avec d'autres langages de programmation, tels que Java, C#, .NET et PHP.

Public cible

Ce cours s'adresse à tout développeur.

Prérequis

Il est recommandé de suivre au préalable les cours « Git Fondamentaux » et « DevOps ».

- Git - Fondamentaux («GITFO»)
- DevOps Foundation («DEVFOK»)

Formations complémentaires

- Docker Administration et Opérations («DOCKER»)
- Les concepts essentiels de Kubernetes («KUBNET»)

Avez-vous une question ou souhaitez-vous organiser un cours en entreprise ?

Nous vous conseillons volontiers au +41 22 738 80 80 ou romandie@digicomp.ch. Retrouvez toutes les informations détaillées concernant les dates sur www.digicomp.ch/formations-software-engineering/programmation/cours-conception-dapplications-cloud-native