

BUT Informatique : QCM Helm – TD3

Nom et prénom :	
--------------------------	--

Contrôle de compréhension – TD3 : Helm

Rappel : Ce questionnaire est un QCM de validation de la compréhension du TD3 sur Helm. **Calculatrice non autorisée – Aucun document n'est autorisé !**

Les questions ont nécessairement **une bonne réponse et une seule**, sauf si elles font apparaître le symbole ♣ et peuvent alors avoir **plusieurs bonnes réponses**. Une bonne réponse vaut **2 points**, une mauvaise réponse vaut **-1 point**, une absence de réponse vaut **0 point**.

QCM :

Question 1 : Qu'est-ce qu'un Chart Helm ?

- ☐ Un graphique de monitoring des ressources Kubernetes
- ☐ Un package contenant tous les fichiers nécessaires pour déployer une application Kubernetes
- ☐ Un outil de visualisation de cluster
- ☐ Un type de service Kubernetes

Question 2 : Comment appelle-t-on le déploiement d'un Chart dans un cluster Kubernetes ?

- ☐ Un deployment
- ☐ Une instance
- ☐ Une release
- ☐ Un package

Question 3 : Quelle est la structure typique d'un Chart Helm ?

- ☐ Dockerfile, docker-compose.yml, README.md
- ☐ Chart.yaml, templates/, values.yaml, charts/
- ☐ deployment.yaml, service.yaml, ingress.yaml
- ☐ package.json, node_modules/, dist/

Question 4 : À quoi sert le fichier Chart.yaml ?

- ☐ À définir les valeurs des variables du Chart
- ☐ À contenir les manifestes Kubernetes
- ☐ À définir le nom, la version, la description et les dépendances du Chart
- ☐ À stocker les secrets de l'application

Question 5 : Quel est le rôle du dossier templates/ dans un Chart ?

- ☐ Stocker les Charts dépendants
- ☐ Contenir les modèles de manifestes Kubernetes au format YAML
- ☐ Définir les versions du Chart
- ☐ Gérer les logs de déploiement

Question 6 : À quoi sert le fichier `values.yaml` ?

- ☐ À définir les secrets de l'application
- ☐ À lister toutes les releases déployées
- ☐ À contenir les valeurs par défaut pour les variables utilisées dans les templates
- ☐ À spécifier les dépendances du Chart

Question 7 : Quelle syntaxe permet de référencer une variable dans un template Helm ?

- ☐ `${.Values.ma.variable}`
- ☐ `{{ .Values.ma.variable }}`
- ☐ `@Values.ma.variable@`
- ☐ `<%= Values.ma.variable %>`

Question 8 : Quelle commande permet d'installer ou mettre à jour un Chart dans un namespace ?

- ☐ `helm deploy <name> <chart> -namespace <ns>`
- ☐ `helm install <name> <chart> -namespace <ns>`
- ☐ `helm upgrade <name> <chart> -namespace <ns>`
- ☐ `helm apply <name> <chart> -namespace <ns>`

Question 9 : Comment packager un Chart Helm ?

- ☐ `helm build <chart-path>`
- ☐ `helm package <chart-path>`
- ☐ `helm create <chart-path>`
- ☐ `helm zip <chart-path>`

Question 10 : Où sont stockés les Charts dépendants dans l'arborescence d'un Chart ?

- ☐ Dans le dossier `dependencies/`
- ☐ Dans le dossier `charts/`
- ☐ Dans le dossier `templates/`
- ☐ Dans le fichier `Chart.yaml`

Question 11 : Comment spécifier une dépendance dans un Chart Helm ?

- ☐ En ajoutant une section `dependencies:` dans `Chart.yaml`
- ☐ En créant un fichier `dependencies.yaml`
- ☐ En utilisant la commande `helm add dependency`
- ☐ En ajoutant le Chart dans le dossier `node_modules/`

Question 12 : Quel fichier permet de définir des fonctions réutilisables dans un Chart ?

- ☐ `functions.yaml`
- ☐ `_helpers.tpl`
- ☐ `utils.yaml`
- ☐ `common.tpl`

Question 13 ♣ : Quelles commandes permettent de surcharger des valeurs lors du déploiement d'un Chart ?

- ☐ `helm upgrade <name> -f values.yaml`
- ☐ `helm upgrade <name> -set cle.variable='valeur'`
- ☐ `helm upgrade <name> -values custom.yaml`
- ☐ `helm upgrade <name> -override cle.variable='valeur'`

Question 14 : Quelle commande permet de voir l'historique des releases d'un Chart ?

- ☐ `helm logs <name>`
- ☐ `helm history <name>`
- ☐ `helm status <name>`
- ☐ `helm list <name>`

Question 15 ♣ : Quels sont les avantages de Helm pour Kubernetes ?

- ☐ Gestion reproductible des déploiements
- ☐ Paramétrage des manifestes sans modification directe
- ☐ Remplacement complet de kubectl
- ☐ Infrastructure as Code (IaC)

Question ouverte :

Question 16 : Expliquez la différence entre un Chart et une Release dans Helm. Donnez un exemple concret d'utilisation.

Rédigez ci-dessous

☐ 0% ☐ 25% ☐ 50% ☐ 75% ☐ 100%

.....

Question 17 : Pourquoi est-il important de versionner ses Charts Helm ? Donnez deux raisons concrètes.

Rédigez ci-dessous

☐ 0% ☐ 25% ☐ 50% ☐ 75% ☐ 100%

.....

Mise en situation :

Contexte : Déploiement multi-environnements

Vous êtes DevOps dans une entreprise qui développe une application web. Vous devez déployer cette application dans trois environnements différents :

- **Développement** : 1 replica, 512 MB RAM, pas de ressources limitées
- **Staging** : 2 replicas, 1 GB RAM, limites de ressources activées
- **Production** : 5 replicas, 2 GB RAM, limites strictes, monitoring activé

L'application utilise une base de données PostgreSQL qui doit également être déployée avec des configurations différentes selon l'environnement.

Question 18 : Comment utiliseriez-vous Helm pour gérer ces trois environnements différents avec un seul Chart ? Décrivez l'organisation de votre Chart et expliquez comment vous géreriez les différentes configurations.

Rédigez ci-dessous

☐ 0% ☐ 25% ☐ 50% ☐ 75% ☐ 100%

.....

Question 19 : Dans le contexte de la question 18, expliquez comment vous organiserez les dépendances entre votre Chart applicatif et le Chart PostgreSQL. Justifiez votre approche.

Rédigez ci-dessous

☐ 0% ☐ 25% ☐ 50% ☐ 75% ☐ 100%

.....

Question 20 : Vous devez intégrer ce Chart Helm dans un pipeline CI/CD GitLab. Décrivez les étapes principales que vous mettriez en place, de la modification du code jusqu'au déploiement en production.

Rédigez ci-dessous

☐ 0% ☐ 25% ☐ 50% ☐ 75% ☐ 100%

.....

Fin du QCM