M2 Dev Web & Mobile

DevOps Partie 2

Projet

Objectifs du projet

- Faire fonctionner le site www.samplephpwebsite.com via un Docker Compose et des Dockerfile

- Créer une configuration Jenkins afin de packager automatiquement l'application à chaque fois qu'une nouvelle version est disponible (ie: nouveau commit)

- Faire une version par branche git (3 versions successives à faire fonctionner : v1, v2, v3) - Penser aux tags des images Docker

- Attention à ne pas faire de superflu (services, applications ou modules inutiles) mais uniquement le strict nécessaire pour faire fonctionner le site.

- Le projet est à faire SEUL. Aucun partage de structure ou de fichiers ne sera accepté.

- BONUS (3 Points) : Créer un script de testing (avec PHPUnit, Atoum ou tout autre framework de testing PHP) dans le but de vérifier que le site fonctionne correctement

Documents complémentaires

- Archive contenant le répertoire git de l'application contenant les 3 versions
- Fichier de configuration Nginx
- Projet type de base du module précédent

Rendu attendu :

- Un repository GitHub qui contiendra 3 branches, une pour chacune des versions du site dans lequel seront les sources du site (non altérées) et les différents fichiers de configuration nécessaires (Scripts Job DSL, Jenkins Pipeline, Dockerfile, Docker Compose, ...)

- Une fois le travail réalisé, merci de vous rendre sur le portail Sharepoint de l'école dans la section Evaluations (https://estiam.sharepoint.com/Evaluations1/) et de déposer votre devoir.

    - Attention, votre rendu est à effectuer au format ZIP et être nommé de la manière suivante : **PRENOM_NOM_E5DWMOPS2.ZIP**. **Toute copie ne respectant pas cette convention ne sera pas corrigée.**

- Avant de quitter le campus, merci de vous rendre à l'accueil pour vous assurer que la copie est bien présente. En cas de copie non déposée ou de non-respect des consignes, vous serez considéré comme absent et une note de 0 (rattrapable) sera appliquée.

Rappel de commandes git

- changer de branche : git checkout <nom-de-la-branche>
- créer un repository : git init
- créer une branche : git branch <nom-de-la-branche>
- préparer un commit en ajoutant des fichiers : git add <fichiers ou répertoires> ou git add --all pour ajouter tous les fichiers modifiés depuis le dernier commit
- commiter : git commit -m "<message du commit>"

Procédure de test

- Création d'un job Jenkins par branche
- Chaque job exécutera un script Job DSL "samplephpwebsite.groovy"

Points bonus

- Documentation propre, claire et détaillée, voire en MarkDown - 2 points

- Ajout d'étapes supplémentaires dans les jobs Jenkins, comme par exemple : linter, code quality, ... (cette liste n'est pas exhaustive et vous pouvez rajouter d'autres types de tâches selon vos envies/idées) - 3 points

- Scan automatique par Jenkins d'un repository contenant des Jenkinsfile - 3 points