|  |  |  |
| --- | --- | --- |
|  |  |  |
| *Nom de naissance* |  | DIA |
| *Nom d’usage* |  | DIA |
| *Prénom* |  | Lauboudou |
| *Adresse* |  | 13 rue Olympe de Gouges 94400 Vitry sur Seine |
|  |  |

|  |  |
| --- | --- |
| **Titre professionnel visé** | |
|  | |
| Administrateur DevOps | |
|  | |
| **Modalité d’accès :** | |
|  | |
|  | Parcours de formation |
|  | Validation des Acquis de l’Expérience (VAE) |
|  | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Présentation du dossier** | | | |
|  | | | |
|  | | | |
|  | | | |
|  | Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel. **Ce titre est délivré par le Ministère chargé de l’emploi.**  Le DP appartient au candidat. Il le conserve, l’actualise durant son parcours et le présente **obligatoirement à chaque session d’examen**.  Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.  Il est consulté par le jury au moment de la session d’examen. | |  |
|  | | | |
| **Pour prendre sa décision, le jury dispose :** | | | |
| 1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l’entretien professionnel ou de l’entretien technique ou du questionnement à partir de productions. 2. du **Dossier Professionnel** (DP) dans lequel le candidat a consigné les preuves de sa pratique professionnelle 3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d’un parcours de formation 4. de l’entretien final (dans le cadre de la session titre).   *[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels*  *du ministère chargé de l’Emploi]* | | | |
| **Ce dossier comporte :** | | | |
| * pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ; * un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d’un titre, d’un diplôme, d’un certificat de qualification professionnelle (CQP) ou des attestations de formation ; * une déclaration sur l’honneur à compléter et à signer ; * des documents illustrant la pratique professionnelle du candidat (facultatif) * des annexes, si nécessaire. | | | |
| *Pour compléter ce dossier, le candidat dispose d’un site web en accès libre sur le site.* | | | |
|  | | [**http://travail-emploi.gouv.fr/titres-professionnels**](http://travail-emploi.gouv.fr/titres-professionnels) | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sommaire** | | | |
| **Exemples de pratique professionnelle** | | | |
| **La mise en place d’un projet DevOps – pipeline CI/CD/CD** | | **p.** | **5** |
|  |  DevOps-CI-CD-Project - Installation de l’infrastructure avec Terraform | p. | 5 |
|  |  DevOps-Project-Ansible – Installation et configuration des outils DevOps | p. | 10 |
|  |  Le déploiement du conteneur Sonarqube-db, SonarQube et Jenkins | p. | 15 |
|  |  La mise en place d’un pipeline sur Jenkins | p. | 21 |
|  |  |  |  |
| **Le déploiement de l’application zodiacjs sur Minikube** | | **p.** | **27** |
|  |  Le déploiement des services sur Minikube | p. | 27 |
|  |  Intitulé de l’exemple n° 2  p. | p. |  |
|  |  Intitulé de l’exemple n° 3  p | p. |  |
|  |  |  |  |
| **Scripting Windows : La mise en place d’un projet Hyper-V** | | **p.** | **34** |
|  |  Création et configuration de Switchs, VM et clusters | p. | 34 |
|  |  Intitulé de l’exemple n° 2  p. | p. |  |
|  |  Intitulé de l’exemple n° 3  p | p. |  |
|  |  |  |  |
| **Intitulé de l’activité-type n° 4** | | **p.** |  |
|  |  Intitulé de l’exemple n° 1  p. | p. |  |
|  |  Intitulé de l’exemple n° 2  p. | p. |  |
|  |  Intitulé de l’exemple n° 3  p | p. |  |
|  |  |  |  |
| **Titres, diplômes, CQP, attestations de formation** *(facultatif)* | | p. | 38 |
| **Déclaration sur l’honneur** | | p. | 39 |
| **Documents illustrant la pratique professionnelle** *(facultatif)* | | p. | 40 |
| **Annexes** *(Si le RC le prévoit)* | | p. | 41 |

**Exemples de pratique**

**professionnelle**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Activité-type** | **1** | | | **La mise en place d’un projet DevOps – pipeline CI/CD/CD** | | | | |
| ***Exemple n°1***  | | | | [***DevOps-CI-CD-Project***](https://github.com/lauboudou/DevOps-CI-CD-Project) ***- Installation de l’infrastructure avec Terraform*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Dans le cadre d’un travail pratique (TP) visant à mettre en place un projet DevOps réalisant un pipeline CI/CD/CD pour assurer l’intégration (CI), le déploiement (CD), le test et la livraison continue (CD) du projet.  Pour la mise en place de ce projet, il est essentiel de préparer et configurer un environnement de travail DevOps sécurisé. L’utilisation de la plateforme Docker est évidant dans ce cas et étant sur un PC portable windows11, j’ai configuré et utilisé différents outils pour la mise en place de cette plateforme.  Dans ce TP, les deux conteneurs créés seront manipulés comme de deux machines virtuelles ubuntu.  Faute d’avoir accès à un Cloud pour toute la promotion, le choix était acté sur la manipulation de conteneurs docker comme des VM. Pour mon projet final, j’utilise de vraies machines virtuelles sur le Cloud d’AWS.   1. **DevOps-CI-CD-Project (Infrastructure as code (IaC))**    * 1. **Création de l’image docker ubuntu-ssh**   J’ai pris le choix de déployer une infrastructure avec terraform sur ma machine windows11, j’ai créé une image docker **ubuntu-ssh** à partir d’un fichier Dockerfile de l'image **fredericeducentre/ubuntu-ssh** préalablement préparée avec docker, ssh, un utilisateur ssh appartenant au groupe root et sudoers et expose le port 22 (port ssh par défaut). Dans ce Dockerfile j’ajoute l'installation des outils sshpass, terraform et ansible.   * + 1. **Création de deux conteneurs**   J’ai configuré et mis en place un projet terraform pour la création automatique de mes deux conteneurs **ubuntu** avec l’image **ubuntu-ssh** créée précédemment sur le même réseau (docker network) pour permettre l’accès et les échanges entre les deux conteneurs en toute sécurité via le protocole ssh.  Les deux conteneurs créés porteront les noms, respectivement **vm-ubuntu-install** et **vm-ubuntu-DevOps**.  Le projet terraform de création des instances DevOps-CI-CD-Project est disponible sur le github à l’url :  [**https://github.com/lauboudou/DevOps-CI-CD-Project.git**](https://github.com/lauboudou/DevOps-CI-CD-Project.git)  Le fichier README.md reprend en détails la mise en place de l’infrastructure.    **Le fichier Dockerfile**    Pour une meilleure sécurité la configuration terraform utilise des variables (voir variables.tf, versions.tf) et automatise la création de deux conteneurs avec des noms différents comme visible dans la boucle affichée sur le fichier main.tf  **Le fichier variables.tf**    **Le fichier main.tf de configuration**    L’accès aux deux conteneurs par ssh avec l’utilisateur test créé dans le Dockerfile, par l’adresse IP du conteneur / port 22 ou localhost / le port mappé  Pour le conteneur vm-ubuntu-install  🡺 ssh test@IP\_conteneur -p 22 ou ssh test@localhost -p 6023  Pour le conteneur vm-ubuntu-DevOps 🡺 ssh test@IP\_conteneur -p 22 ou ssh test@localhost -p 6024 | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| **Windows11**  **WSL**  **Docker Desktop**  **Docker**  **Terraform**  **GitHub**  **DockerHub**  **SSH**  **VS Code**  **Recherche sur la documentation officielle de** Docker, Dockerhub et Terraform **pour comprendre l’utilisation des commandes et la configuration à mettre en place**  <https://docs.docker.com/docker-hub/>  https://developer.hashicorp.com/terraform?product\_intent=terraform | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| **J’ai réalisé ce TP en autonomie.** | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***Formation DevOps 2024 – G2R*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | **Module Terraform** | | | | | |
| **Période d’exercice** | |  | **Du** | | **30/05/2024** | | **au** | **31/05/2024** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| **J’ai travaillé sur ce projet de TP que pendant le temps libre, autrement je suivais des modules dans la journée.** | | | | | | | | |

|  |  |  |
| --- | --- | --- |
| **Activité-type** | **1** | **La mise en place d’un projet DevOps – pipeline CI/CD/CD** |
| ***Exemple n°2***  | | ***DevOps-Project-Ansible – Installation et configuration des outils DevOps*** |
|  | |  |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | |
| Le projet DevOps-Project-Ansible contient les playbooks et les inventories ansible configurés pour la création des outils DevOps que j’ai utilisé dans cette activité.  Le projet DevOps-Project-Ansible est disponible sur le github à l’url :  <https://github.com/lauboudou/DevOps-Project-Ansible.git>    Le sous projet DevOps-Project sera copié dans le conteneur **vm-ubuntu-install** configuré dans l’exemple n° 1 via le playbook ansible surligné en jaune dans l’image ci-dessus **copy-config-playbook.yml** il sera lancé avec le fichier **inventaire.yml**  Le fichier **copy-config-playbook.yml**    Le fichier **inventory.yml**      Note : le fichier **inventory.yml** est en clair. Pour la sécurité il sera crypté avec un mot de passe qui sera demandé à chaque lancement du playbook.  A la racine de ce projet lancer  ansible-playbook -i inventory.yml copy-config-playbook.yml --ask-become-pass --ask-vault-pass  Les credentials seront demandés :  become-pass= le passeword du root  vault-pass= le password    Le résultat est une copie du projet **DevOps-Project** qui sera déposé dans dans le répertoire /home/test  Le projet 'DevOps-Project' sera utilisé pour configurer le conteneur **vm-ubuntu-DevOps**  Dans le dossier **/home/test/DevOps-Project/config-vm-ubuntu-devops** contient :   * Le fichier **playbook.yml** pour installer jenkins, sonardb (postgres) et sonnarqube. * Le fichier **playbook-agent-node.yml** pour installer l'agent node agent\_reactjs\_node pour le bon fonctionnement d'un projet reactJS   En occurence l'agent "fredericeducentre/jenkins\_agent\_node" utilisé pendant les cours sera utilisé pour ce TP  Le fichier **inventaire.yml** contient la configuration du conteneur **vm-ubuntu-DevOps** pour accéder en ssh.  Ce fichier inventory.yml est crypté par la commande : ansible-vault encrypt inventaire.yaml  Il sera décrypté après le mot de passe demandé par **vault-pass** à l’exécution d’un playbook.  Les **playbook.yml** et **playbook-agent-node.yml** du dossier config-vm-ubuntu-devops utiliseront cet **inventory.yml** crypté | | |
|  | | |
| **2. Précisez les moyens utilisés :** | | |
| **GitHub**  **SSH**  **Ansible**  **VS Code** | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | |
| **J’ai réalisé ce TP en autonomie.** | | | | | | |
|  | | | | | | |
| **4. Contexte** | | | | | | |
|  | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | ***Formation DevOps 2024 – G2R*** | | |
|  |  | | | | | |
| **Chantier, atelier, service** |  | **Module Ansible** | | | | |
| **Période d’exercice** |  | **Du** | **03/06/2024** | | **au** | **07/06/2024** |
|  | | | | | | |
|  | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | |
| **J’ai travaillé sur ce projet de TP que pendant le temps libre, autrement je suivais des modules dans la journée.** | | | | | | |

|  |  |  |
| --- | --- | --- |
| **Activité-type** | **1** | **La mise en place d’un projet DevOps – pipeline CI/CD/CD** |
| ***Exemple n°3***  | | ***Le déploiement du conteneur Sonarqube-db, SonarQube et Jenkins*** |
|  | |  |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | |
| L’objectif de cette partie, c’est la mise en place d’une plateforme DevOps permettant l’exécution d’un pipeline **CI** (Intégration Continue) / **CD** (Déploiement Continue) / **CD** (Livraison Continue)  Le serveur Jenkins accompagné par un serveur SonarQube et sa base des données Sonarqube-db permettront la mise en place de cette plateforme.  Pour l’installation de ces serveurs conteneurisés, je me connecte sur le conteneur **vm-ubuntu-install**  Dans le dossier /home/test/DevOps-Project/config-vm-ubuntu-devops  Je veux lancer le **playbook.yml** qui va se connecter en ssh sur le conteneur **vm-ubuntu-devops** et y installer les conteneurs **jenkins**, **sonarqube-db** et **sonarqube**.   1. Dans un premier temps, lancer la commande suivante pour installer la clé ssh sur **vm-ubuntu-install** avant d'exécuter le **playbook.yml** en précisant l'@IP de ==> vérifier l'@IP de votre conteneur par docker inspect **vm-ubuntu-devops**   sudo ssh test@IP\_vm-ubuntu-devops -p 22  [alt text](https://github.com/lauboudou/DevOps-CI-CD-Project/blob/main/images/image-6.png)  Taper exit pour sortir du conteneur.  Dans le dossier **/home/test/DevOps-Project/config-vm-ubuntu-devops** le fichier **inventory.yml** est configuré avec l'adresse IP du conteneur **vm-ubuntu-devops**   1. Créer l'image **dlaubo/sonarqube** pour l'installation de **sonarqube** nous avons besoin de mettre à jour l'image récupérer sur dockerhub.com avant de l'utiliser pour contourner l'exception suivante constatée pendant le test **quality** de **sonarqube**   Exception in thread "Thread-0" java.lang.NoClassDefFoundError:  **ch/qos/logback/classic/spi/ThrowableProxy**  Toujours dans le conteneur **vm-ubuntu-install** mettez-vous dans le dossier  **/home/test/DevOps-Project/dockerfile-sonarqube**  Un fichierDockerfile est présent nous allons créer l'image **dlaubo/sonarqube** par la commande suivante  docker build . -t dlaubo/sonarqube  Je m’assure que l'image est créée et réviens dans le dossier  **/home/test/DevOps-Project/config-vm-ubuntu-devops**  Maintenant j’exécute le playbook.yml comme suit,  ansible-playbook -i inventory.yml playbook.yml --ask-become-pass --ask-vault-pass  J’assure que l'installation est bien passée sur **vm-ubuntu-DevOps**  Les serveurs sont installés : jenkins sur le port 8080 et sonnarqube sur le port 9000  Aller se le connecter jenkins ==> <http://localhost:8080/>  Le mot de passe demandé se trouvera dans le fichier /var/\*\*\* indiqué sur l’interface qui s’affiche Renseigner le mot de passe et continuer la configuration  Une fois l’installation terminée, le serveur est accessible comme suit,  [alt text](https://github.com/lauboudou/DevOps-CI-CD-Project/blob/main/images/image-7.png)  [alt text](https://github.com/lauboudou/DevOps-CI-CD-Project/blob/main/images/image-8.png)  Configuer **l'agent node** sur jenkins Aller dans Administrer Jenkins > Nodes >  Créer un nouveau node : **agent\_reactjs\_node** passer le nombre d'exécution sur le node controlleur à 0  Dans le conteneur **vm-ubuntu-install** Aller dans le répertoire /home/test/DevOps-Project/config-vm-ubuntu-devops  Récupérer le token d'accès de l'agent node sur jenkins: Cliquer sur le nom du node agent fraîchement créé et copier le token séléctionné sur l'image.    Aller dans Tableau de bord > Administrer Jenkins > Nœuds  [image](https://github.com/lauboudou/DevOps-CI-CD-Project/blob/main/images/image9.png)  Coller le token en remplacement du token présent après jenkis\_container:8080 dans le fichier **/home/test/DevOps-Project/config-vm-ubuntu-devops/playbook-agent-node.yml**  http://jenkins\_container:8080 <Coller ici le token récupéré>  Sauvegarder et rester dans le répertoire : **/home/test/DevOps-Project/config-vm-ubuntu-devops**  lancer le playbook-agent-node.yml pour installer l'agent node **agent\_reactjs\_node**  ansible-playbook -i inventory.yml playbook-agent-node.yml --ask-become-pass --ask-vault-pass  L'agent jenkins\_agent\_node sera installé, assurer qu'il soit démarré et vérifier le sur l'interface de jenkins    Configurer sonnarqube sur l'interface [**http://localhost:9000/**](http://localhost:9000/)  Une fois terminée, la page d’accueil sera comme suit    Configurer sonarqube sur jenkins en ajoutant ses plugins  Aller dans Tableau de bord > Administrer Jenkins > Plugins    Redémarrer le serveur jenkins | | |
|  | | |
| **2. Précisez les moyens utilisés :** | | |
| **GitHub**  **SSH**  **Docker 🡺 docker build pour la création d’une image**  **Ansible**  **Jenkins**  **Sonarqube** | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | |
| **J’ai réalisé ce TP en autonomie.** | | | | | | |
|  | | | | | | |
| **4. Contexte** | | | | | | |
|  | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | ***Formation DevOps 2024 – G2R*** | | |
|  |  | | | | | |
| **Chantier, atelier, service** |  | **Module Jenkins** | | | | |
| **Période d’exercice** |  | **Du** | **10/06/2024** | | **au** | **12/06/2024** |
|  | | | | | | |
|  | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | |
| **J’ai travaillé sur ce projet de TP que pendant le temps libre, autrement je suivais des modules dans la journée.** | | | | | | |

|  |  |  |
| --- | --- | --- |
| **Activité-type** | **1** | **La mise en place d’un projet DevOps – pipeline CI/CD/CD** |
| ***Exemple n°4***  | | ***La mise en place d’un pipeline sur jenkins*** |
|  | |  |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | |
| Tester l'exécution d'un pipeline pour confirmer le bon fonctionnement de la plateforme mise en place.  J’ai produit spécialement pour ce test, un projet **reactjs** pour pouvoir utiliser le agent\_reactjs\_node que j’ai configuré sur jenkins.  Le projet reactjs est accessible sur github à l’url : [**https://github.com/lauboudou/zodiacJS.git**](https://github.com/lauboudou/zodiacJS.git)  Aller sur l’interface sonarqube  Créer un nouveau projet    Configurer ce projet de test en local sur sonarqube, générer et récupérer le token    Une configuration comme celle-ci sera affichée  -Dsonar.projectKey=Test-horoscope-zodiac-js -Dsonar.sources=. -Dsonar.host.url=[http://172.20.0.4:9000](http://172.20.0.4:9000/) -Dsonar.token=sqp\_\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ''' } }  Qui nous servira dans le pipeline jenkins,  Allons sur l’interface jenkins et configurer un pipeline :  Tableau de bord > Nouveau Item  Renseigner le nom et sélectionner le type : Pipeline  Cliquer sur OK    Dans la partie pipeline script écrire ce pipeline  pipeline {  agent {  label 'agent\_reactjs\_node'  }  /\*environment{  DOCKER\_HUB\_PAT = credentials('docker\_hub\_pat')  }\*/  stages {  stage('clone') {  steps {  git branch: 'main', credentialsId: 'PAT\_Git', url: 'https://github.com/lauboudou/zodiacJS.git'  }  }  stage('build') {  steps {  sh 'npm install && npm run build'  }  }  stage('test') {  steps {  sh 'npm run test'  }  }  stage('delivery') {  steps {  sh 'docker login -u dlaubo -p ${DOCKER\_HUB\_PAT}'  sh 'docker build . -t dlaubo/zodiacjs:${BUILD\_ID}'  sh 'docker push dlaubo/zodiacjs:${BUILD\_ID}'  }  }  }  post {  success {  echo 'This will run only if successful'  mail bcc: 'admin2@admin.com', body: '''Bonjour, Si vous recevez ce mail, c\'est OK pour moi ! from zodiacJS Cdt ''', cc: 'admin1@admin.com', from: '', replyTo: '', subject: 'Jenkins Test Email', to: 'admin@admin.com'  }  failure {  mail bcc: '', body: "<b>Failure Example</b><br>Project: ${env.JOB\_NAME} <br>Build Number: ${env.BUILD\_NUMBER} <br> URL de build: ${env.BUILD\_URL}", cc: '', charset: 'UTF-8', from: '', mimeType: 'text/html', replyTo: '', subject: "ERROR CI: Project name -> ${env.JOB\_NAME}", to: "admin@admin.com"  }  }  }  Les stages du pipeline sont :   * Stage Clone : pour accéder à github et récupérer le git repository, utilisera le credential PAT d’accès à un repo privé configuré au préalable sur jenkins 🡺 CI * Stage build : pour le déploiement du projet et le copier dans le répertoire du travail /var/jenkins\_home/workspace/Test\_Zodiac\_ReactJS 🡺 CD * Stage test : pour exécuter les tests unitaires disponibles * Stage delivery : le build de création de l’mage docker du projet et sa livraison sur dockerhub 🡺 CD * Post : pour l’envoi d’un mail de **success** ou de **failure** selon le résultat de l’exécution du pipeline   Sauvegarder et exécuter le pipeline.  Si le pipeline stage view n'est pas visible penser à vérifier si le plugin pipeline stage view est installé sinon installer le et redémarrer jenkins. Retester le pipeline.    Le pipeline est exécuté correctement.  Aller sur la sortie de la console pour voir en détail l’exécution de chaque étape du pipeline.    Nous trouvons dans les logs que chaque stage du pipeline est exécuté :   * Stage Clone : le git repository est récupéré 🡺 CI * Stage build : le projet reactjs est déployé dans le répertoire du travail du projet /var/jenkins\_home/workspace/Test\_Zodiac\_ReactJS 🡺 CD * Stage test : le projet est testé, dans ce cas des tests unitaires sont lancés * Stage delivery : l’mage docker du projet est créée et livrée sur dockerhub 🡺 CD * Post : un mail de success est envoyé car le pipeline est exécuté correctement   L’image dlaubo/zodiacjs est bien délivré sur dockerhub    Vérifier le mail envoyé sur **http:mailtrap.io** (le serveur mail sur jenkins est configuré avec mailtrap) | | |
|  | | |
| **2. Précisez les moyens utilisés :** | | |
| **Jenkins**  **Sonarqube**  **GitHub**  **Docker 🡺 docker build et push pour la création et la livraison de l’image**  **Mailstrap**  **Npm**  **Node js** | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **3. Avec qui avez-vous travaillé ?** | | | | | | |
| **J’ai réalisé ce TP en autonomie.** | | | | | | |
|  | | | | | | |
| **4. Contexte** | | | | | | |
|  | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | ***Formation DevOps 2024 – G2R*** | | |
|  |  | | | | | |
| **Chantier, atelier, service** |  | **Module Jenkins** | | | | |
| **Période d’exercice** |  | **Du** | **17/06/2024** | | **au** | **21/06/2024** |
|  | | | | | | |
|  | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | |
| **J’ai travaillé sur ce projet de TP que pendant le temps libre, autrement je suivais des modules dans la journée.** | | | | | | |

|  |  |  |
| --- | --- | --- |
| **Activité-type** | **2** | **Le déploiement de l’application zodiacjs sur Minikube** |
| ***Exemple n°1***  | | ***Le déploiement des services sur Minikube*** |
|  | |  |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | |
| Minikube c’est un environnement de tests facile d’installation et de configuration et permet de créer un cluster kubernetes et d’avoir un nœud unique sur une machine locale ou sur un environnement de développement. Cette offre d’environnement de tests rapides était le choix pour la mise en place de cette activité.  Dans cette activité, j’ai mis en place et déployé des services sur l’environnement de minikube,  **Les prérequis pour cette activité**   * Avoir WSL et docker installés * Installer minikube sur WSL en suivant le lien : <https://minikube.sigs.k8s.io/docs/start/> * Démarrer minikube   Le projet utiliser pour cette activité est accessible à l’url :  [**https://github.com/lauboudou/horoscope-zodiac-js.git**](https://github.com/lauboudou/horoscope-zodiac-js.git)    **Pour déployer cette application ‘horoscope-zodiac-js’ sur minikube, j’ai suivi les étapes suivantes :**  A la racine de ce projet, l’image docker est créée: conteneurisez votre application **horoscope-zodiac-js** à l’aide de Dockerfile ci-dessous    docker build . -t -**horoscope-zodiac-js:latest**  L’image **horoscope-zodiac-js:latest** sera créée  Cette création envoie l’image Docker au démon Docker de Minikube : cela permet à Minikube d’accéder à l’image.  Les fichiers YAML de déploiement et de service Kubernetes : définissez les ressources Kubernetes nécessaires au déploiement de votre application.      Déployer le service sur minikube : utilisez kubectl pour appliquer vos configurations Kubernetes.  **Voici les étapes détaillées** :  Démarrez minikube s’il n’est pas en cours d’exécution  minikube Start  **Assurez-vous d’exécuter ce projet dans l’environnement minikube**  minikube -p minikube docker-env | Invoke-Expression  **Exécuter les fichiers yaml**  kubectl apply -f deployment.yaml  kubectl apply -f service.yaml  Accéder à l’application **horoscope-zodiac-js**  Pour accéder à votre application, vous devez obtenir l’URL du service :  minikube service **horoscope-zodiac-js-service** –url  Cette commande renverra une URL que vous pouvez ouvrir dans votre navigateur pour voir votre application React.js déployée.  Déployez **horoscope-ingress.yaml** et activez le contrôleur d’entrée    Ajoutez d’abord cette configuration (hostname) en tant qu’admin dans le fichier 'C :\Windows\System32\drivers\etc\hosts'  127.0.0.1 **site.horoscope**  **Déployer l’entrée d’horoscope**  kubectl apply -f **horoscope-ingress.yaml**  **Activer l’entrée**  Les addons minikube permettent l’entrée  **Afficher l’entrée disponible**  kubectl get ingress  [texte alternatif](https://github.com/lauboudou/horoscope-zodiac-js/blob/main/src/images/image1.png)  **Démarrer l’entrée de l’horoscope**  Minikube tunnel  **Sur une autre fenêtre shell voici quelques commandes**  kubectl get pods 🡺 unité de service cohérent  kubectl get deployment  kubectl get service  **Vérifier l’accessibilité de l’application d’horoscope sur le navigateur**  /site.horoscope  [texte alternatif](https://github.com/lauboudou/horoscope-zodiac-js/blob/main/src/images/image2.png)  La page d’accueil est accessible. | | |
|  | | |
| **2. Précisez les moyens utilisés :** | | |
| **Jenkins**  **Sonarqube**  **GitHub**  **Docker 🡺 docker build et push pour la création et la livraison de l’image**  **Mailstrap**  **Npm**  **Node js** | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **3. Avec qui avez-vous travaillé ?** | | | | | | |
| **J’ai réalisé ce TP en autonomie.** | | | | | | |
|  | | | | | | |
| **4. Contexte** | | | | | | |
|  | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | ***Formation DevOps 2024 – G2R*** | | |
|  |  | | | | | |
| **Chantier, atelier, service** |  | **Module Kubernetes (K8S)** | | | | |
| **Période d’exercice** |  | **Du** | **24/06/2024** | | **au** | **28/06/2024** |
|  | | | | | | |
|  | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | |
| **J’ai travaillé sur ce projet de TP que pendant le temps libre, autrement je suivais des modules dans la journée.** | | | | | | |

|  |  |  |
| --- | --- | --- |
| **Activité-type** | **3** | **Scripting Windows : La mise en place d’un projet Hyper-V** |
| ***Exemple n°1***  | | ***Création et configuration de Switchs, VM et clusters*** |
|  | |  |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | |
| L’objectif de ce projet et la mise en place d’une infrastructure robuste, disponible et performant sur windows11 en utilisant le scripting windows et hyper-v.  En se basant sur l’architecture du projet Hyper V ci-dessous (voir l’image) j’ai créé les VMs, les switchs virtuels, les OU, les Groups, les utilisateurs et les services à l’aide des commandes sur PowerShell.  Le site web qui sera déployé sur cette infrastructure devrait gérer une grande quantité des données et nécessite d’être disponible 24h/24h et 7j/7j d’où la répartition des charges (load balancing) sur le cluster NLB.   Pour le bon fonctionnement de ce projet et avant tout Hyper-V est activé sur windows11  * Toutes la création et configuration des éléments de ce projet sont exécutées sur powershell.  La création des Switchs Virtuels  |  |  | | --- | --- | | Nom du Switch | Type du Switch | | MPIO1 | Privé | | MPIO2 | Privé | | PULSATION | Privé | | INTERNE | Interne | | EXTERNE | Externe |  Exemple des commandes de créationCréation des switchs New-VMSwitch -name MPIO1 -SwitchType Private  New-VMSwitch -name MPIO2 -SwitchType Private  New-VMSwitch -name PULSATION -SwitchType Private  New-VMSwitch -name INTERNE -SwitchType Internal  New-VMSwitch -name EXTERNE -NetAdapterName wi-Fi  Les nouvelles connexions    Création des disques virtuels  New-VHD -ParentPath C:\Hyper-V\Master\Master.vhdx -Path C:\Hyper-V\DC-01\DC-01.vhdx -Differencing  New-VHD -ParentPath C:\Hyper-V\Master\Master.vhdx -Path C:\Hyper-V\Hote-01\Hote-01.vhdx -Differencing  New-VHD -ParentPath C:\Hyper-V\Master\Master.vhdx -Path C:\Hyper-V\Hote-02\Hote-02.vhdx -Differencing  New-VHD -ParentPath C:\Hyper-V\Master\Master.vhdx -Path C:\Hyper-V\Hote-03\Hote-03.vhdx -Differencing    Les VMs créées sont visibles sur Hyper-V    Ces commandes sont à titre d’exemple. Ce projet est détaillé dans le **Rendu du TP** joint à ce dossier professionnel.  **Entre autres les actions détaillées sont** :  La fixation de l’adresse IP, la mise en place de DNS, DHCP, configuration des utilisateurs, des groupes, des OU, des pools de stockage, des disques virtuels dans les VMs, des VMs embriquées,  Les disques virtuels dans les VMs 🡺 3 disques seront utilisés par le load balancing et un disque comme Témoin    Les VMs embriquées VM **SRV-WEB1** et **SRV-WEB2** | | |
|  | | |
| **2. Précisez les moyens utilisés :** | | |
| **Windows11**  **Hyper-V**  **PowerShell** | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| **J’ai réalisé ce TP en autonomie.** | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | |  | | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | ***Formation DevOps 2024 – G2R*** | | | |
|  |  | | | | | | | |
| **Chantier, atelier, service** |  | **Module Scripting Windows** | | | | | | |
| **Période d’exercice** |  | **Du** | | **10/04/2024** | | **au** | **07/06/2024** | |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| **J’ai travaillé sur ce projet de TP que pendant le temps libre, autrement je suivais des modules dans la journée.** | | | | | | | | |
| **Titres, diplômes, CQP, attestations de formation** | | | | | | | | |
|  | | | | | | | | |
| *(facultatif)* | | | | | | | | |
| **Intitulé** | | | **Autorité ou organisme** | | | | | **Date** |
| **PMI-PMP Project Management Professional** | | | **Actinuum** | | | | | **28/12/2018** |
| **Devenir Manager Agile** | | | **Actinuum** | | | | | **17/04/2019** |
| **Formation Ingénieur Concepteur et Administrateur de bases de données en Système Ouvert** | | | **Université de Versailles - IUT de Vélizy** | | | | | **27/05/2005** |
| **Master / DESS Ingénieur des réseaux et des systèmes** | | | **Université de Versailles** | | | | | **03/09/2001** |
| Cliquez ici. | | | Cliquez ici pour taper du texte. | | | | | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | | | Cliquez ici pour taper du texte. | | | | | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | | | Cliquez ici pour taper du texte. | | | | | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | | | Cliquez ici pour taper du texte. | | | | | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | | | Cliquez ici pour taper du texte. | | | | | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | | | Cliquez ici pour taper du texte. | | | | | Cliquez ici pour sélectionner une date. |

|  |
| --- |
| **Déclaration sur l’honneur** |
|  |
|  |

*Lauboudou DIA*

Je soussigné(e) [prénom et nom] ,

déclare sur l’honneur que les renseignements fournis dans ce dossier sont exacts et que je suis l’auteur(e) des réalisations jointes.

*Vitry sur Seine*

19/09/2024

Fait à le

pour faire valoir ce que de droit.

Signature :

|  |
| --- |
| **Documents illustrant la pratique professionnelle** |
|  |
| *(facultatif)* |
| **Intitulé** |
| **TP Projet Hyper V** |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
| **Annexes** |
|  |
| *(Si le RC le prévoit)* |
|  |