Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER PROFESSIONNEL (DP)

Nom de naissance	DA SILVA MARCELINO
Nom d'usage	Entrez votre nom d'usage ici.
Prénom	Nuno Ricardo
Adresse	7Ter Rue Pierre Lavoye 95300 PONTOISE

Titre professionnel visé

Administrateur Systémes DevOps

MODALITE D'ACCES :

☒ 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

Sommaire

Exemples de pratique professionnelle

EXEMPLES DE PRATIQUE PROFESSIONNELLE

Activité-type 1

Automatiser le déploiement d'une infrastructure dans le Cloud

Exemple n°1 ▸ Automatiser la création de serveurs à l'aide de scripts

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

1. Définition des ressources d'infrastructure dans des fichiers Terraform : J'ai créé des fichiers Terraform décrivant l'infrastructure à déployer dans le cloud AWS, y compris les instances EC2, les groupes de sécurité, les réseaux VPC, etc.
2. Configuration d'ArgoCD : J'ai configuré ArgoCD pour surveiller les changements dans un référentiel Git où sont stockés les fichiers Terraform. ArgoCD est utilisé pour détecter les modifications et déclencher le déploiement automatique de l'infrastructure dans AWS.
3. Développement de scripts de démarrage EC2 : J'ai écrit des scripts shell ou des scripts de configuration (par exemple, utilisant Ansible) pour automatiser le processus de configuration des instances EC2 nouvellement créées. Ces scripts peuvent inclure l'installation de logiciels, la configuration des paramètres système, etc.
4.

2. Précisez les moyens utilisés :

AWS : J'ai utilisé les services AWS, notamment EC2, VPC, IAM, etc., pour créer et gérer l'infrastructure cloud.
Terraform : J'ai utilisé Terraform pour décrire de manière déclarative l'infrastructure à déployer dans le cloud AWS. Terraform permet de gérer l'infrastructure en tant que code et fournit une méthode efficace pour automatiser le déploiement.
ArgoCD : J'ai configuré ArgoCD pour automatiser le déploiement de l'infrastructure décrite dans les fichiers Terraform. ArgoCD surveille les modifications dans le référentiel Git et déploie automatiquement les changements détectés.
Scripts de démarrage EC2 : J'ai développé des scripts (par exemple, des scripts shell, des playbooks Ansible, etc.) pour automatiser la configuration des instances EC2 après leur création. Ces scripts garantissent que les instances sont prêtes à l'emploi dès leur démarrage, sans intervention manuelle supplémentaire.

Exemple n°2 ▸ **Automatiser le déploiement d'une infrastructure**

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Définition de l'infrastructure dans des fichiers Terraform : Création de fichiers Terraform décrivant les ressources nécessaires dans AWS, telles que les instances EC2, les bases de données, les réseaux VPC, les groupes de sécurité, etc.
Configuration d'ArgoCD : Mise en place d'ArgoCD pour surveiller un référentiel Git contenant les fichiers Terraform. ArgoCD est utilisé pour détecter les modifications et déclencher le déploiement automatique de l'infrastructure dans AWS.
Paramétrage des pipelines CI/CD : Création de pipelines CI/CD pour automatiser le processus de test, de construction et de déploiement des applications sur l'infrastructure nouvellement créée.
Gestion des secrets : Intégration d'un système de gestion des secrets, tel que AWS Secrets Manager ou HashiCorp Vault, pour stocker et récupérer de manière sécurisée les informations sensibles nécessaires au déploiement de l'infrastructure.

2. Précisez les moyens utilisés :

Terraform : Utilisation de Terraform pour définir de manière déclarative l'infrastructure comme code. Terraform permet une gestion efficace et reproductible des ressources cloud.
AWS : Utilisation des services AWS, tels que EC2, RDS, VPC, IAM, etc., pour créer et gérer l'infrastructure cloud.
ArgoCD : Configuration d'ArgoCD pour automatiser le déploiement de l'infrastructure décrite dans les fichiers Terraform. ArgoCD assure un déploiement cohérent et fiable de l'infrastructure à chaque modification.
Outils CI/CD : Utilisation d'outils de CI/CD tels que Jenkins, GitLab CI/CD, AWS CodePipeline ou Github Actionsn pour automatiser les processus de test, de construction et de déploiement des applications sur l'infrastructure cloud.
Système de gestion des secrets : Intégration d'un système de gestion des secrets pour sécuriser les informations sensibles telles que les clés d'accès API, les mots de passe de base de données, etc., nécessaires au déploiement de l'infrastructure.

Exemple n°3 ▸ **Sécuriser l'infrastructure**

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Configuration des politiques IAM : Création de politiques IAM pour limiter les privilèges des utilisateurs et des services AWS. Cela inclut l'attribution de permissions spécifiques aux rôles et aux utilisateurs afin de limiter l'accès aux ressources critiques.
Utilisation de Vault pour la gestion des secrets : Intégration de HashiCorp Vault pour stocker et distribuer de manière sécurisée les informations sensibles telles que les clés d'accès API, les mots de passe et les jetons d'authentification.
Gestion des secrets avec GitHub : Utilisation des fonctionnalités de gestion des secrets de GitHub pour stocker de manière sécurisée les informations sensibles telles que les clés SSH, les jetons d'API, etc., nécessaires aux déploiements et aux interactions avec les services GitHub.
Configuration d'Iptables pour Linux : Utilisation d'Iptables pour configurer et maintenir des règles de pare-feu sur les instances Linux. Cela permet de contrôler le trafic réseau entrant et sortant et de renforcer la sécurité en limitant l'accès aux ports et aux services.
Activation de l'authentification à double facteur (2FA) : Activation de l'authentification à double facteur pour les comptes AWS et GitHub. Cela ajoute une couche de sécurité supplémentaire en exigeant une deuxième méthode d'authentification, comme un code généré ou une notification sur un appareil mobile, en plus du nom d'utilisateur et du mot de passe.

2. Précisez les moyens utilisés :

AWS IAM : Utilisation des services IAM d'AWS pour définir et gérer les politiques d'accès aux ressources AWS. IAM permet de contrôler de manière granulaire les actions qu'un utilisateur ou un service peut effectuer sur les ressources AWS.
HashiCorp Vault : Intégration de Vault pour la gestion des secrets, en utilisant ses fonctionnalités de stockage sécurisé et de distribution des secrets. Vault offre des mécanismes avancés de chiffrement et de contrôle d'accès pour garantir la sécurité des informations sensibles.
GitHub : Utilisation des fonctionnalités de gestion des secrets de GitHub pour stocker de manière sécurisée les informations sensibles utilisées dans les déploiements et les interactions avec les services GitHub.
Iptables : Configuration et gestion d'Iptables sur les instances Linux pour contrôler le trafic réseau et renforcer la sécurité en limitant l'accès aux services autorisés.
Authentification à double facteur : Activation de l'authentification à double facteur pour les comptes AWS et GitHub, en utilisant les fonctionnalités intégrées de ces plateformes pour renforcer la sécurité des comptes utilisateur.

Exemple n°4 ▸ Mettre l'infrastructure en production dans le Cloud

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Test et validation : Effectuer des tests approfondis sur l'infrastructure déployée pour garantir son bon fonctionnement et sa conformité aux exigences de sécurité.
Gestion des secrets : Intégrer les secrets nécessaires au déploiement dans l'infrastructure de production en utilisant Vault pour le stockage sécurisé et la distribution des informations sensibles.
Configuration d'Iptables : Mettre en place et optimiser les règles Iptables sur les instances Linux pour renforcer la sécurité et contrôler le trafic réseau entrant et sortant.
Activation de l'authentification à double facteur : Assurer que l'authentification à double facteur est activée pour tous les comptes AWS et GitHub utilisés dans l'environnement de production pour renforcer la sécurité des accès.
Surveillance continue : Mettre en place des outils de surveillance pour surveiller en permanence l'état de l'infrastructure en production, détecter les anomalies et répondre rapidement aux incidents éventuels.

2. Précisez les moyens utilisés :

Tests automatisés : Utilisation de scripts de tests automatisés pour vérifier le bon fonctionnement de l'infrastructure déployée, assurant ainsi sa stabilité et sa fiabilité.
Intégration de Vault : Intégration de HashiCorp Vault pour gérer de manière sécurisée les secrets nécessaires au déploiement de l'infrastructure en production, garantissant ainsi la confidentialité des informations sensibles.
Configuration et optimisation d'Iptables : Utilisation de scripts et de configurations Iptables pour mettre en place des règles de pare-feu robustes et sécurisées sur les instances Linux en production.
Activation de l'authentification à double facteur : Configuration des paramètres de sécurité pour activer l'authentification à double facteur sur les comptes AWS et GitHub, garantissant ainsi une couche de sécurité supplémentaire pour les accès aux ressources.
Outils de surveillance : Mise en place d'outils de surveillance tels que AWS CloudWatch, Prometheus, Grafana, etc., pour surveiller les métriques système, les journaux d'application et les performances de l'infrastructure en temps réel, assurant ainsi une réactivité optimale aux incidents et aux problèmes potentiels.

3. Avec qui avez-vous travaillé ?

1. **Les formateurs** :
   - Ce sont les experts qui dirigent les sessions de formation, présentent les concepts, répondent à vos questions et vous guident dans les travaux pratiques et les challenges.
   - Ils fournissent des retours d'expérience, des conseils et des orientations pour vous aider à progresser dans votre parcours de formation.
2. **Les autres apprenants** :
   - Ce sont vos camarades de classe qui suivent la même formation que vous.
   - Vous travaillez en collaboration avec eux lors des travaux pratiques, des challenges et des sessions de travail en groupe.
   - Vous partagez des idées, des connaissances et des expériences pour vous entraider et surmonter les défis rencontrés pendant la formation.

4. Contexte

Nom de l'entreprise, organisme ou association	***Ecole O'Clock***
Chantier, atelier, service	**Ecole o'clock**
Période d'exercice	**Du** 23/01/2024 **au** 05/07/2024

5. Informations complémentaires *(facultatif)*

Cliquez ici pour taper du texte.

Activité-type 2

Déployer en continu une application

Exemple n°1 ▸ Préparer un environnement de test

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Création d'un référentiel GitHub/GitLab : Créer un référentiel GitHub pour héberger le code source de l'application Django et de l'application Node.js.
Configuration des tests automatisés : Écrire des tests automatisés pour l'application Django (par exemple, des tests unitaires et des tests d'intégration) et pour l'application Node.js (par exemple, des tests Jest). Configurer ces tests pour qu'ils s'exécutent automatiquement lors de chaque modification du code.
Définition d'un pipeline CI/CD : Configurer un pipeline CI/CD à l'aide de GitHub Actions ou enkins pour automatiser le processus de test et de déploiement de l'application. Le pipeline doit inclure des étapes telles que la construction des applications, l'exécution des tests et le déploiement sur l'environnement de test.
Préparation de l'environnement de test : Créer une infrastructure pour l'environnement de test, y compris les serveurs nécessaires pour exécuter l'application Django et l'application Node.js. Configurer également les bases de données et autres services requis pour l'environnement de test.
Déploiement continu : Mettre en place un mécanisme de déploiement continu qui déploie automatiquement les nouvelles versions de l'application sur l'environnement de test dès qu'elles passent avec succès les tests.

2. Précisez les moyens utilisés :

GitHub Actions : Utiliser GitHub Actions pour définir et exécuter le pipeline CI/CD, en automatisant les tests et le déploiement de l'application à chaque modification du code.
Tests automatisés : Écrire des tests automatisés pour les applications Django et Node.js à l'aide de frameworks de test appropriés comme Django's TestCase pour Django et Jest pour Node.js. Configurer ces tests pour qu'ils s'exécutent automatiquement dans le pipeline CI/CD.
Infrastructure as Code (IaC) : Utiliser des outils comme Terraform ou AWS CloudFormation pour définir et gérer l'infrastructure de l'environnement de test de manière reproductible et automatisée.
Services cloud : Utiliser des services cloud comme AWS, Google Cloud Platform (GCP) ou Microsoft Azure pour provisionner et gérer les ressources nécessaires à l'environnement de test, y compris les serveurs, les bases de données et autres services.
Configuration management : Utiliser des outils de gestion de configuration comme Ansible ou ou avec le Runner Actions pour automatiser la configuration des serveurs de l'environnement de test, en garantissant une configuration cohérente et reproductible.

Exemple n°2 ▸ Gérer le stockage des données

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Configuration de AWS S3 : Création de compartiments (buckets) dans AWS S3 pour stocker les données d'objets. Configurer les politiques de versioning et de sécurité appropriées pour garantir l'intégrité et la confidentialité des données.
Déploiement de bases de données SQL et NoSQL sous Linux ou Docker : Installer et configurer les bases de données SQL (par exemple, MySQL, PostgreSQL) et NoSQL (par exemple, MongoDB, Redis) sur des instances Linux ou des conteneurs Docker.
Planification des sauvegardes : Élaborer un plan de sauvegarde pour les bases de données, déterminant la fréquence des sauvegardes complètes et incrémentielles, ainsi que les stratégies de rétention des sauvegardes.
Mise en place de sauvegardes automatisées : Créer des scripts de sauvegarde qui utilisent les outils appropriés (comme mysqldump pour MySQL ou mongodump pour MongoDB) pour sauvegarder régulièrement les données des bases de données. Planifier l'exécution de ces scripts à l'aide d'un outil de planification de tâches tel que cron.
Redondance géographique : Configurer la réplication des données dans des régions AWS distinctes pour garantir une redondance géographique en cas de défaillance dans une région. Utiliser les fonctionnalités de réplication multi-région d'AWS S3 pour les données d'objets, ainsi que les mécanismes de réplication intégrés aux bases de données pour les données structurées ou non structurées.

2. Précisez les moyens utilisés :

AWS S3 : Utiliser AWS S3 pour le stockage sécurisé et évolutif des objets. Configurer les compartiments et les politiques de sécurité selon les besoins spécifiques du projet.
Bases de données SQL et NoSQL : Déployer des bases de données SQL et NoSQL sur des serveurs Linux ou dans des conteneurs Docker. Configurer les paramètres de sécurité et de performance pour répondre aux exigences de l'application.
Scripts de sauvegarde : Écrire des scripts de sauvegarde qui utilisent les outils de sauvegarde appropriés pour chaque type de base de données. Automatiser l'exécution de ces scripts à l'aide de l'outil de planification de tâches cron.
Planification des sauvegardes : Élaborer un plan de sauvegarde détaillé, en définissant la fréquence des sauvegardes, les stratégies de rétention et les procédures de restauration en cas de besoin.
Réplication multi-région : Configurer la réplication multi-région pour les données critiques stockées dans AWS S3 et dans les bases de données, assurant ainsi une redondance géographique et une haute disponibilité en cas de défaillance dans une région. Utiliser les fonctionnalités intégrées de réplication et de failover pour garantir la continuité des opérations en cas d'incident.

Exemple n°3 ▸ Gérer des containers

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Gestion des conteneurs avec Docker :

- Création de fichiers Dockerfile pour définir les environnements d'exécution des applications.
- Utilisation de Docker pour construire des images conteneurisées à partir des Dockerfiles.
- Déploiement et gestion de conteneurs individuels à l'aide de Docker sur des machines virtuelles Linux.
- Utilisation de Docker Compose pour définir et orchestrer des applications multi-conteneurs.
- Configuration de volumes Docker pour le stockage persistant des données.

Orchestration des conteneurs avec Kubernetes :

- Configuration d'un cluster Kubernetes sur des machines virtuelles AWS EC2.
- Déploiement d'applications conteneurisées sur le cluster Kubernetes à l'aide de fichiers de configuration YAML (pods, services, déploiements, etc.).
- Utilisation de fonctionnalités avancées de Kubernetes telles que l'auto-scaling, les mises à jour automatiques, et le contrôle de la répartition de charge.
- Gestion des ressources Kubernetes telles que les nœuds, les pods, les services, etc.
- Mise en place de stratégies de sauvegarde et de restauration pour les données stockées dans les conteneurs Kubernetes.

2. Précisez les moyens utilisés :

Pour la gestion des conteneurs avec Docker :

- Utilisation de Docker Engine pour la création et la gestion des conteneurs sur des machines virtuelles Linux.
- Déploiement de Docker Compose pour orchestrer des applications multi-conteneurs sur des machines virtuelles.
- Configuration de volumes Docker pour le stockage persistant des données.
- Utilisation de fichiers Dockerfile pour définir les environnements d'exécution des applications conteneurisées.

Pour l'orchestration des conteneurs avec Kubernetes :

- Configuration d'un cluster Kubernetes sur des machines virtuelles AWS EC2 en utilisant kops, kubeadm ou EKS (Amazon Elastic Kubernetes Service).
- Utilisation de fichiers de configuration YAML pour déployer des applications conteneurisées sur le cluster Kubernetes.
- Gestion des ressources Kubernetes via l'interface en ligne de commande (kubectl) ou l'interface graphique Kubernetes Dashboard.

Exemple n°4 ▸ Automatiser la mise en production d'une application avec une plateforme

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Configuration du pipeline CI/CD : Définir un pipeline CI/CD qui automatisera les étapes de test, de construction et de déploiement de l'application.
Intégration avec le système de contrôle de version : Connecter le pipeline CI/CD à votre système de contrôle de version (comme GitHub, GitLab, Bitbucket) pour déclencher automatiquement le processus de déploiement lorsqu'il y a des modifications dans le code source.
Tests automatisés : Ajouter des tests automatisés pour valider la qualité du code avant le déploiement en production. Cela peut inclure des tests unitaires, des tests d'intégration, des tests de performance, etc.
Construction de l'application : Créer un artefact exécutable de l'application à déployer. Cela peut être un fichier JAR, un conteneur Docker, ou tout autre format adapté à votre environnement d'exécution.
Déploiement automatique : Configurer le pipeline CI/CD pour déployer automatiquement l'application sur l'environnement de production une fois les tests réussis et l'artefact construit.

2. Précisez les moyens utilisés :

Outil de déploiement continu : Utiliser une plateforme de déploiement continu telle que Jenkins, GitLab CI/CD, ou AWS CodePipeline pour créer et gérer le pipeline CI/CD.
Système de contrôle de version : Intégrer le pipeline CI/CD avec votre système de contrôle de version (GitHub, GitLab, Bitbucket) pour déclencher les déploiements automatiquement en cas de nouvelles modifications du code source.
Outils de tests automatisés : Utiliser des frameworks de test automatisé tels que JUnit, Selenium, Jest, ou PHPUnit pour écrire et exécuter des tests automatisés qui valident la qualité et le bon fonctionnement de l'application.
Outils de construction d'application : Utiliser des outils de construction d'application tels que Maven, Gradle, ou Docker pour générer des artefacts exécutables de l'application prêts à être déployés.
Outils de déploiement : Utiliser des outils de déploiement automatisé tels que Ansible, Terraform, ou des scripts de déploiement personnalisés pour automatiser le déploiement de l'application sur l'environnement de production.

3. Avec qui avez-vous travaillé ?

1. **Les formateurs** :
   - Ce sont les experts qui dirigent les sessions de formation, présentent les concepts, répondent à vos questions et vous guident dans les travaux pratiques et les challenges.
   - Ils fournissent des retours d'expérience, des conseils et des orientations pour vous aider à progresser dans votre parcours de formation.
2. **Les autres apprenants** :
   - Ce sont vos camarades de classe qui suivent la même formation que vous.
   - Vous travaillez en collaboration avec eux lors des travaux pratiques, des challenges et des sessions de travail en groupe.
   - Vous partagez des idées, des connaissances et des expériences pour vous entraider et surmonter les défis rencontrés pendant la formation.

4. Contexte

Nom de l'entreprise, organisme ou association	▸	***Ecole O'Clock***
Chantier, atelier, service	▸	**Ecole o'clock**
Période d'exercice	▸	**Du** 23/01/2024 **au** 05/07/2024

5. Informations complémentaires *(facultatif)*

Cliquez ici pour taper du texte.

Activité-type 3

Superviser les services déployés

Exemple n°1 ▸ Définir et mettre en place des statistiques de services

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Configuration de Prometheus : Déploiement et configuration de Prometheus pour collecter et stocker des métriques sur les services déployés. Cela inclut la définition des cibles de surveillance (endpoints des services à superviser) dans la configuration de Prometheus.
Instrumentation des applications : Instrumentation des applications à l'aide de bibliothèques ou de middlewares Prometheus pour exposer des métriques HTTP. Ces métriques peuvent inclure des informations sur les performances, les erreurs et d'autres statistiques pertinentes pour la supervision des services.
Définition d'alertes : Configuration des règles d'alerte dans Prometheus pour détecter les problèmes potentiels ou les défaillances des services. Ces règles d'alerte peuvent être basées sur des seuils de métriques spécifiques ou sur des conditions plus complexes définies par des expressions de requête PromQL.
Intégration avec Grafana : Intégration de Prometheus avec Grafana pour visualiser les métriques collectées et créer des tableaux de bord personnalisés pour surveiller les performances des services. Grafana offre une interface conviviale pour explorer les données et créer des visualisations informatives.

2. Précisez les moyens utilisés :

Prometheus : Déploiement et configuration de Prometheus pour collecter et stocker des métriques. Configuration des cibles de surveillance dans le fichier de configuration Prometheus pour indiquer les endpoints des services à superviser.
Instrumentation des applications : Utilisation de bibliothèques ou de middlewares Prometheus pour instrumenter les applications et exposer des métriques HTTP. Ces métriques sont collectées par Prometheus pour la supervision des services.
Définition d'alertes : Configuration des règles d'alerte dans Prometheus pour détecter les problèmes potentiels ou les défaillances des services. Les règles d'alerte sont définies à l'aide du langage de requête PromQL et peuvent être personnalisées en fonction des besoins spécifiques de supervision.
Intégration avec Grafana : Configuration de Grafana pour se connecter à Prometheus et visualiser les métriques collectées. Création de tableaux de bord personnalisés dans Grafana pour surveiller les performances des services et détecter les anomalies.

Exemple n°2 ▸ Exploiter une solution de supervision

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Déploiement de Prometheus : Installer et configurer Prometheus sur votre infrastructure. Vous pouvez le déployer sur un serveur dédié ou sur un cluster Kubernetes, selon votre environnement.
Configuration des cibles de surveillance : Définir les cibles de surveillance dans le fichier de configuration de Prometheus. Cela peut inclure les endpoints des services à surveiller, les bases de données, les serveurs, etc.
Instrumentation des applications : Instrumenter vos applications pour exposer des métriques Prometheus. Cela peut être fait en ajoutant des bibliothèques ou des middlewares Prometheus à vos applications existantes, ou en utilisant des exporteurs Prometheus pour des applications tierces.
Définition des règles d'alerte : Configurer des règles d'alerte dans Prometheus pour détecter les anomalies ou les pannes potentielles. Ces règles peuvent être basées sur des seuils de métriques ou sur des conditions plus complexes définies par des expressions PromQL.
Intégration avec d'autres outils : Intégrer Prometheus avec d'autres outils de supervision comme Grafana pour la visualisation des données, ou des outils de notification comme Alertmanager pour la gestion des alertes.

2. Précisez les moyens utilisés :

Déploiement de Prometheus : Utilisation des outils de déploiement comme Helm pour Kubernetes ou des scripts d'installation pour déployer Prometheus sur votre infrastructure.
Configuration des cibles de surveillance : Édition du fichier de configuration Prometheus YAML pour définir les cibles de surveillance. Vous pouvez également utiliser des services de découverte de service pour découvrir automatiquement les cibles à surveiller.
Instrumentation des applications : Utilisation de bibliothèques Prometheus pour instrumenter vos applications et exposer des métriques. Vous pouvez également utiliser des exportateurs Prometheus pour des applications tierces qui ne supportent pas nativement Prometheus.
Définition des règles d'alerte : Configuration des règles d'alerte dans le fichier de configuration Prometheus YAML. Les règles d'alerte sont définies en utilisant le langage de requête PromQL pour interroger les métriques et détecter les anomalies.
Intégration avec d'autres outils : Configuration de l'intégration entre Prometheus et d'autres outils comme Grafana ou Alertmanager. Vous pouvez utiliser des configurations spécifiques dans les fichiers de configuration Prometheus pour définir ces intégrations.

Exemple n°3 ▸ Echanger sur des réseaux professionnels éventuellement en anglais

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

LinkedIn : LinkedIn est la plateforme professionnelle la plus populaire. Vous pouvez rejoindre des groupes liés à votre domaine d'activité, suivre des entreprises et des leaders d'opinion, et interagir avec d'autres professionnels du monde entier.
Twitter : Twitter est également une excellente plateforme pour se tenir au courant des dernières actualités et tendances dans votre domaine. Vous pouvez suivre des experts et des organisations pertinentes et participer à des discussions en utilisant des hashtags spécifiques.
Reddit : Reddit héberge de nombreux sous-reddits dédiés à différents domaines professionnels et technologiques. Vous pouvez trouver des communautés actives où vous pouvez poser des questions, partager des articles et discuter de sujets pertinents.
Stack Overflow : Stack Overflow est principalement connu comme une plateforme de questions-réponses pour les développeurs, mais il dispose également de sections dédiées aux discussions sur les carrières, les technologies et les bonnes pratiques professionnelles.
Meetup : Meetup est une plateforme qui permet de découvrir et de participer à des événements professionnels locaux dans votre domaine d'intérêt. Vous pouvez rencontrer des personnes partageant les mêmes idées et élargir votre réseau professionnel.

2. Précisez les moyens utilisés :

GitHub : GitHub est la plus grande plateforme de développement collaboratif au monde. Vous pouvez explorer des projets open source, contribuer à des projets existants et interagir avec d'autres développeurs via les discussions et les problèmes.
DevOps Forum : Les forums DevOps comme DevOps.com ou Reddit/r/DevOps sont des endroits où les professionnels partagent des idées, des expériences et des conseils sur les pratiques DevOps, les outils et les technologies.
TechCrunch : TechCrunch est un site Web d'actualités technologiques qui couvre les dernières tendances, les startups émergentes et les développements dans le monde de la technologie. Vous pouvez également trouver des forums de discussion liés aux articles sur le site.

3. Avec qui avez-vous travaillé ?

1. **Les formateurs** :
   - Ce sont les experts qui dirigent les sessions de formation, présentent les concepts, répondent à vos questions et vous guident dans les travaux pratiques et les challenges.
   - Ils fournissent des retours d'expérience, des conseils et des orientations pour vous aider à progresser dans votre parcours de formation.
2. **Les autres apprenants** :
   - Ce sont vos camarades de classe qui suivent la même formation que vous.
   - Vous travaillez en collaboration avec eux lors des travaux pratiques, des challenges et des sessions de travail en groupe.
   - Vous partagez des idées, des connaissances et des expériences pour vous entraider et surmonter les défis rencontrés pendant la formation.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ ***Ecole O'Clock***

Chantier, atelier, service ▸ **Ecole o'clock**

Période d'exercice ▸ **Du** 23/01/2024 **au** 05/07/2024

5. Informations complémentaires *(facultatif)*

Cliquez ici pour taper du texte.

Titres, diplômes, CQP, attestations de formation

(facultatif)

Intitulé	Autorité ou organisme	Date
BEP/CAP électrotechnique	**LP Claude Chappe (Nanterre)**	**30/07/1999**
DÉVELOPPEUR APPLICATIONS PYTHON Bac +4 titre RNCP	**Openclassrooms**	**30/05/2021**
Administrateur système DevOps Bac +4 titre RNCP	**O'clock**	**05/07/2024**
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.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.

Déclaration sur l'honneur

Je soussigné(e) [prénom et nom] *Nuno Ricardo Da Silva Marcelino*,

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

Fait à *Pontoise* le 06/03/2024

pour faire valoir ce que de droit.

Signature :

Documents illustrant la pratique professionnelle

(facultatif)

Intitulé
Cliquez ici pour taper du texte.

ANNEXES

(Si le RC le prévoit)