Co-travail Dev – Ops
« Au bon beurre »
V2.0

Inclut parties 2 et 3

Préambule :

Rendu :
Maquette opérationnelle présentant les services et les technologies demandées. Un dossier d'architecture présentant les éléments, leur imbrication et les configurations est aussi demandé. Les fichiers de scripts commentés seront joints ainsi que les copies d'écran que vous pensez nécessaires. Une archive comprendra l'ensemble des éléments.

En cas d'apprenants en nombre impair, l'arbitrage sera fait par l'intervenant.

Objectifs pédagogiques :
L'objectif de ces scénarios est d'amener le binôme à mettre en commun les compétences de développement, de réseaux, d'infrastructure d'outil ainsi que de gestion de projet (équipes, compétences, urgent / important). Les deux axes principaux du DevOps sont abordés : intégration multi compétences et intégration des processus de développement et de production

Attention :
Un certain nombre de technologies demandées doivent faire l'objet de recherches sur internet et de travail personnel pour arriver à un niveau compatible avec les attentes.

Les technologies identifiées sont obligatoires (scénario d'un historique existant) et non pas accessoires. La notation prendra en compte ces éléments.

N'oubliez pas que vous êtes des professionnels de l'informatique et que vos solutions doivent être suffisamment maitrisées et documentées pour permettre la pérennité de votre activité et la transmission aisée de votre savoir à vos futurs collaborateurs.

Scénario :

Le Client, société mono site employant 350 personnes pour un chiffre d'affaires annuel de 65 millions d'euros élabore et fabrique des produits laitiers à partir de lait collecté par les coopératives de la région. Ces produits sont frais et juste pasteurisés. On y retrouve différents types de laits, beurre, fromages type feta, mozzarella, préparations apéritives. Ces produits sont très sensibles notamment aux bactéries (salmonelles) et aux contaminants éventuels de l'environnement (produits d'entretien, polluants atmosphériques, restes de fermentation, rongeurs, insectes, activité humaine ...).

L'entreprise ayant entamé une démarche « Food Defense » par obligation réglementaire récente, elle se doit de contrôler strictement chaque étape et lancer des alertes en cas de suspicion de contamination. Statistiquement, les actes malveillants sont internes à 70%. Un problème non détecté à temps pourrait être dramatique pour la santé des consommateurs et l'image de la société. Ses expéditions seraient purement et simplement interdites.

Les mesures techniques (confinement, automatisation, contrôle d'accès, surveillance vidéo et périmétrique, analyses de la chaine du froid, bactérienne et chimique …) ont déjà été implantées. La visualisation des paramètres ne se fait cependant que sur les pupitres de commandes et il est très difficile à ce jour de centraliser l'information. Le délai de réactivité en cas d'événement est beaucoup trop long pour être compatible avec les exigences sanitaires. Les lots incriminés seraient purement et simplement détruits, occasionnant ainsi de très grosses pertes financières.

Ces différents paramètres de métrologies sont issus des 5 unités d'exploitation distinctes (types de produits, procédés, paramètres) représentant un total de 6000 points de mesure par unité et par heure (soit 100 paramètres relevés 60 fois par heure). Ces points de mesures sont remontés par 15 types d'automates différents dotés de modules de communication spécifiques et parfois « exotiques ».

Certains de ces automates sont très anciens et nécessitent d'adapter des formats d'échange propriétaires mal documentés. Ce sera à prendre en compte dans le projet en se basant sur l'hypothèse que des interfaces techniques exposant les paramètres au format Json sur socket TCP/IP ont été réalisées par une société tierce pour les 2/3 des automates. Mais le tiers restant rencontre des difficultés de débits, de pertinence des valeurs remontées et de stabilité nécessitant encore de nombreux réglages. Un traitement à 6 mois pour le dernier tiers est annoncé à ce jour par le sous-traitant.

Le client possède une infrastructure locale de serveurs informatiques techniquement proche de la votre ce qui a été déterminant dans l'attribution du marché.

Votre mission :

Le contrat de prestation a été signé pour la réalisation d'un intranet permettant l'édition de tableaux de bords temps réel largement paramétrables afin de visualiser les éléments de suivi de la chaine de traitement des aliments.

L'ensemble des paramètres de chaque unité doit être intégré en étant clairement identifié (quelle unité, quel automate, quels paramètres, état de maintenance de

l'automate …). Plusieurs vues seront proposées pour détailler ou non les paramètres observés en fonction des profils utilisateurs et des éléments de contexte.

L'ergonomie devra être adaptée à une surveillance par 2 opérateurs de permanence visualisant l'ensemble des unités de production.

L'interface Web utilisateur devra être claire, lisible, fluide, imagée avec une lecture très graphique permettant cependant de fouiller les chiffres en cas de d'analyse poussée par l'opérateur. Ces graphiques seront exportables dans des formats PDF signés à des fins de constitution de dossier de traçabilité à destination de l'inspection sanitaire (volet réglementaire contrôlé par un organisme externe certifié et assermenté pouvant requérir des sanctions). Les logs de mesure pourront être extraits à la demande et insérés dans le dossier PDF.

Les règles métiers de calcul des métriques sont fournies par le client sous forme de logigrammes réglementaires et il fournit aussi des jeux de test normés (données brutes -> données transformées) afin de pouvoir contrôler les traitements réalisés par le site intranet et s'assurer que les calculs sont bons. Les valeurs de tolérance sont aussi fixées mais devront pouvoir évoluer en fonction de la réglementation. Tout changement dynamique des paramètres sera tracé et authentifié. Cela fera l'objet d'une validation à blanc à chaque livraison majeure et en fin de projet.

La base de données sera gérée avec différents profils utilisateurs ainsi que des droits très précis et restrictifs afin de garantir sa robustesse et la traçabilité de toute action en toute circonstance (principe de moindre privilège). La destruction ou l'altération malveillantes de la base devront être quasi impossibles de part cette gestion fine et les moyens de redondance technique à définir. Au vu de la densité des données collectées une architecture robuste permettant de ne pas perdre de données devra être mise en place avec justification des choix. Ces choix seront présentés au client dans les premières phases du projet. Une sauvegarde régulière devra être organisée selon les règles de l'art.

Les communications entre les modules d'adaptation des automates (format Json via socket TCP/IP) et le serveur Web devront être sécurisées via chiffrement et VLAN dédiés.

Un gros effort doit être réalisé sur la qualité et la profondeur des tests.

Livrables du projet.

Partie 1

Seuls les éléments suivants sont à réaliser dans cette première approche. Le sujet sera complété au fur et à mesure et les réalisations devront donc être consolidées d'une étape à l'autre.

Automates :

Chaque unité de production produira toute les minutes (valeur à rendre configurable pour la démonstration) un fichier Json unique comportant des informations fixes ou générées aléatoirement et du type suivant pour chacun des automates (10 par unité) :

		Type de données	Valeurs possibles
Numéro d'unité	entier	fixe	1 à 5
Numéro d'automate	entier	fixe	1 à 10
Type d'automate	entier	fixe	Un couple numéro d'automate et type d'automate est défini au début du travail Types : de 0X0000BA20 à de 0X0000BA2F
Température cuve	float	aléatoire	Entre 2,5 et 4 degrés par dixième de degré
Température extérieure	float	aléatoire	Entre 8 et 14degrés par dixième de degré
Poids du lait en cuve	float	Aléatoire	Entre 3512 kg et 4607kg par incrément de 1 kilo
Poids du produit fini réalisé	float	Aléatoire	différence entre les deux mesures successives de poids du lait dans la cuve
Mesure pH	float	Aléatoire	Entre 6,8 et 7,2 par incrément de 1/10
Mesure K+	entier	Aléatoire	Entre 35mg et 47 mg /litres par pas de 1 mg
concentration de NaCl	float	Aléatoire	Entre 1g et 1,7 g part litre par pas de 0,1g
Niveau bactérien salmonelle	entier	Aléatoire	Entre 17 et 37 ppm , par incrément de 1 ppm
Niveau bactérien E-coli	entier	Aléatoire	Entre 35 et 49 ppm , par incrément de 1 ppm
Niveau bactérien Listéria	entier	Aléatoire	Entre 28 et 54 ppm , par incrément de 1 ppm

Exemple : Les 10 automates de l'unité 2 produisent chacun toutes les minutes les mesures précédentes qui sont intégrées dans un seul et même fichier Jason renseigné à la date courante et mis à disposition du système.

Son nom sera constitué de :
« paramunite »_<numéro unité>< « _ »><date unix epoch> « .json »

Ce programme de génération aléatoire est réalisé impérativement en Python 3.7.x .

Hypothèses : les données sont considérées comme bonnes une fois renseignées dans le fichier Json. Dans cette étape, aucun contrôle supplémentaire ne sera réalisé, alors vérifiez bien la qualité des données.

Récupération des données:

Les données issues de la collecte de capteurs seront insérées dans une base de données MariaDB, à l'aide de scripts automatisés.

La communication entre le mécanisme de lecture des fichiers json et de génération se fera par socket TCP/IP chaque fois qu'un fichier sera disponible.

La sécurité de la base de données n'est pas traitée dans cet incrément.

Exploitation des données :

Les données sont affichées graphiquement sur un serveur Web exploitant les données en base, avec un rafraichissement toutes les minutes. Dans cette première version,

l'amplitude d'observation est de 60 minutes max sur une fenêtre glissante. i.e les données les plus anciennes antérieures à T-60 minutes ne sont pas affichées.

Plate-forme :

Vous mettrez à disposition un environnement conteneurisé sur la base d'une Debian 9.9 composé d'au moins 5 conteneurs Dockers représentant chacun une unité de production conformément au scénario. Chaque docker exécutera les scripts de génération de mesure et pourra communique en socket TCP/IP avec le gestionnaire de base de données en MariaDB.

Le serveur Web sera réalisé à partir de nginx et présentera graphiquement et dynamiquement les résultats. Réalisation des écrans à partir du Langage Javascript.

Les codes sources et les fichiers de configurations sont gérés sur GitHub.

Le langage de scripting est python sauf pour la partie serveur Web , codé en javascript.

Bonus pour la notation : un orchestrateur Kubernetes permettra de gérer les instances de conteneurs

Partie 2

Mise en œuvre de la solution d'intégration continue Jenkins.

Le principe est le suivant :

A partir du repository GitHub qui contient vos sources python et Javascript, mettre en place une solution Jenkins dans un docker afin de réaliser les tâches d'intégration continue.

https://jenkins.io/doc/pipeline/tour/getting-started/

Un ensemble de tests d'intégration basiques sera décrit dans une étape de test pour vérifier la qualité du code. La solution sera déployée automatiquement.

Réalisation

Mettre en place la solution Jenkins (attention mise à jour Java à prévoir)

- *« A machine with:*
  - *256 MB of RAM, although more than 512MB is recommended*
  - *10 GB of drive space (for Jenkins and your Docker image)*
- *The following software installed:*
  - *Java 8 (either a JRE or Java Development Kit (JDK) is fine)*
  - *Docker (navigate to **Get Docker** at the top of the website to access the Docker download that's suitable for your platform) «*

- *Download and run Jenkins*

- *Download Jenkins.*
- *Open up a terminal in the download directory.*
- *Run java -jar jenkins.war --httpPort=8080.*
- *Browse to http://localhost:8080.*
- *Follow the instructions to complete the installation.*

- *When the installation is complete, you can start putting Jenkins to work!*

Interconnecter avec GitHub.

Créer les pipelines (jobs) de récupération des sources sur le repository github.

https://jenkins.io/doc/tutorials/build-a-python-app-with-pyinstaller/

Réaliser un ensemble de tests basiques à sur la version en ajoutant une étape de tests
Créer la phase de déploiement en ajoutant une étape de delivery.

Faire de même pour la partie Javascript :
https://jenkins.io/doc/tutorials/build-a-node-js-and-react-app-with-npm/

Rendu :

Présenter l'architecture et les Jenkins files.

Réaliser un scénario qui sur modification d'un code dans GitHub, construit l'application python ou Javascript et la déploie.

Partie 3

Cette partie constitue le projet BDD de fin d'année.

Elle s'appuie sur les deux parties précédentes. La présentation finale doit comporter la synthèse des étapes initiales intégrées aux fonctionnalités de la partie 3.

Elle se focalise spécifiquement sur la base de données avec les principes suivants :

Modèle de données :

Le modèle de données doit être mis à jour pour faciliter la montée en puissance (rajouter facilement des unités de productions, des sites et des types d'automates) et gérer efficacement les utilisateurs avec les profils et les éléments de sécurité.

Précisez et argumentez vos choix

Un schéma du modèle de données (Entité Relation) sera présenté.

Les données par défaut, si besoin, son définies comme suit.

donnée	type	Valeur par défaut
Type d'automate	entier	0XFFFFFFFF
Température cuve	float	99,99
Température extérieure	float	99,99
Poids du lait en cuve	float	9999999,9
Poids du produit fini réalisé	float	9999999,9
Mesure pH	float	9999,9
Mesure K+	entier	0XFFFF
concentration de NaCl	float	9999,9
Niveau bactérien salmonelle	entier	0XFFFF
Niveau bactérien E-coli	entier	0XFFFF
Niveau bactérien Listéria	entier	0XFFFF

Les scripts de création des bases seront présentés. La création par interface graphique n'est pas autorisée pour cette partie.

Gestion des droits utilisateurs.

La base de données MariaDB gère les droits des différents utilisateurs :

L'administrateur « root » a accès toutes les fonctionnalités de la base sans restriction. Il délègue un administrateur avec des droits restreints à la base du projet

L' « administrateur délégué », peut créer/détruire les tables pour le projet et y mener toutes les actions courantes. Il attribue les droits aux utilisateurs C'est l'opérateur/adminsigtrateur habituel de la base de ce projet. Il ne voit pas les autres projets.

Le « concentrateur docker » peut lire et écrire dans la base. Il ne peut pas détruire la base, les tables ou les créer. Il ne peut pas créer/détruire/modifier d'utilisateur. Son périmètre se limite aux tables dont il a effectivement besoin.

La partie « DataVision » (graphique) peut lire mais ne peut pas modifier les données enregistrées dans la base, hormis les champs mis à jour automatiquement. Elle ne peut pas détruire la base, les tables ou les créer. Elle ne peut pas créer/détruire/modifier d'utilisateur. Son périmètre se limite aux tables dont elle a effectivement besoin.

Chacun de ces 4 utilisateurs doit être connecté et être authentifié pour accéder à la base.

Créez les profils correspondants et les affecter aux modules concernés.

Sécurité des échanges.

Entre les unités de production (U1 à U5) et le concentrateur docker, un système de chiffrement asymétrique sera mis en œuvre. Le couple clé publique/clé privé pourra par exemple être généré avec OpenPGP/GP4win ou PGPSuite.

Les différentes unités (U1 à U5) représentées par des conteneurs docker chiffrent chaque fichier json avec la clé publique du concentrateur. Celui-ci déchiffre avec sa clé privée et analyse chaque fichier reçu. Il les insert dans la base une fois les tests réalisés.

Le principe de l'analyse de la qualité des données des fichiers json est le suivant :

	Type de données à vérifier	Valeur des données à vérifier	Action à réaliser en cas d'écart
Numéro d'unité	entier	1<=N <= 5	*Insérer les données dans une table d'erreur en attente d'analyse. Un « trigger » est déclenché dans la base sur écriture dans cette table. Il signale par mail à l'administrateur délégué l'erreur avec le date TimeStamp de l'enregistrement et l'unité en cause. Unité 999 si erreur sur l'unité.*
Numéro d'automate	entier	1<= N <=10	
Type d'automate	entier	0X0000BA20<= N <= 0X0000BA2F	
Température cuve	float	0.0<= T <=100.0	
Poids du lait en cuve	float	0.0<=P< 10000.0 kg	

Les échanges de données entre la base, le concentrateur et la DataVision sont chiffrés par SSL/TLS.

Le programme python du concentrateur doit être modifié pour réaliser ces opérations de connexions chiffrées et de contrôle de qualité des données.

Un scénario démontrera le chiffrement des échanges.

Un scenario avec cas d'erreur devra être présenté pour démontrer la fonctionnalité de contrôle de qualité des données et la réaction à une erreur.

Compression et chiffrement de la base :

La base de données doit être chiffrée et compressée notamment avec les fonctions natives de MariaDB.

Un scénario démontrera le chiffrement et la compression de la base et le choix/paramétrage du mode de chiffrement.

Dump de la base :

Créer un script python de sauvegarde de la base de données avec une fréquence quotidienne.

Pour les besoins de la démonstration la date exacte du dump sera configurable pour le créneau de la présentation.

Bonus : Un scenario permettra de lire le contenu de la base dumpée après déchiffrement par l'administrateur délégué.

Rendu :

Outre les réalisations ci-dessus, réaliser un scénario de démonstration qui présente les différents points sur la maquette en accord avec les parties 1 à 3. Tous les choix doivent pouvoir être justifiés et les fichiers de configuration/scripting mis à disposition. Un PPT reprenant les points majeurs sera parcouru.