BUT3 R&T : SAé 5.01

Développer et déployer un microservice dans un environnement virtualisé

1. Objectifs et problématique professionnelle

Le professionnel R&T spécialisé en Développement Système et Cloud, dès lors qu'il intervient dans une équipe de développement, doit pouvoir contribuer à la conception d'applications.

Ces applications s'appuient souvent sur des architectures de microservices : l'application est construite sur la base d'un ensemble de services modulaires.

Ces services modulaires peuvent être déployés de façon indépendante et être mis en œuvre avec souplesse dans des infrastructures virtualisées (abordées dans la SAÉ3.DevCloud.04 "Mettre en place une infrastructure virtualisée").

2. Description générique

Le professionnel DevCloud collabore à des projets de développement d'applications sous forme de microservices. Il apporte ses compétences dans les différentes phases du projet pour :

- développer les composants de l'application ou adapter les composants d'une application fournie pour la faire évoluer vers une architecture n-tiers ;
- développer une (ou des) API permettant les échanges entre les composants de l'application ;
- déployer les composants de l'application sur des serveurs virtualisés ou dans des conteneurs ;
- sécuriser l'application et ses composants, en mettant en place les autorisations et les authentifications adéquates pour les utilisateurs et les clients applicatifs (gestion de session, sécurisation de l'API) et des échanges réseaux (protocoles, ...).

3. Présentation de la SAé

Au cours de cette SAé, vous allez reprendre la **SAE23** "Mettre en place une solution informatique pour l'entreprise" réalisé au S2.

Pour mémoire, ce projet consistait à réaliser une application WEB (`HTML/CSS/PHP/PDO/SQLite/Javascript`) et utilisant les sessions utilisateurs.

Vous allez faire évoluer cette application sous forme de différentes versions en appliquant les concepts évoqués lors des ressources:

- Fondamentaux de la conteneurisation
- Développement de microservices

3.1. Moyens & modalités

- Travail en monome
- 30h de TP en autonomie dont une présentation des résultats en soutenance

> Produire un compte rendu `.md` pour chaque version avec toutes les actions réalisées et les preuves de fonctionnement
>
> Produire les ensembles de fichiers fonctionnels! pour chaque version avec une notice d'installation

3.2. Outils & Langages

- `GitLab` ou `GitHub`: créez vous un compte!
- Travaillez dans une VM Debian Linux fournie sous VMWareWorksation
- Serveur Apache
- Docker
- PHP
- Python (FastAPI, Flask)

3.3. Prérequis:

Commencer par refaire/finir les TPs des ressources BUT2

- Fondamentaux de la conteneurisation
- Développement de microservices

VISA 0

4. Travail organisé en 5 versions

4.1. Version 1: PHP & SQLite

- Reprenez et corrigez votre application afin de la rendre opérationnelle dans la VM.
    - Créez (si nécessaire) une seconde base de données pour les utilisateurs pouvant se connecter avec les droits (admin ou simple utilisateur). Il faut une BDD de données et une BDD pour l'authentification.
- Utilisez GitLab pour piloter le dépôt de vos codes sources.
- Déployez l'application avec `Docker` en utilisant un `Dockerfile`.

VISA 1

4.2. Version 2: PHP & MariaDB

- Modifiez la version 1 en remplaçant `SQLite` par `MariaDB` pour la BDD de données et la BDD authentification.
- Utilisez `GitLab` pour piloter le dépôt de vos codes sources.
- Déployez l'application avec `Docker` en utilisant un `Dockerfile` pour `PHP` et un autre pour `MariaDB`.

VISA 2

4.3. Version 3: PHP & MariaDB & API Python

- Créez une API Python avec `FastAPi` qui permet de réaliser les opérations CRUD sur au moins une table de votre base de données.
- Créez une API Python avec `FastAPi` pour gérer l'authentification simple (sans jeton)
- Modifiez votre programme `PHP` afin de réaliser l'interaction avec les BDD via vos API en utilisant `CURL`.
- Utilisez `GitLab` pour piloter le dépôt de vos codes sources.
- Déployez l'application avec `Docker` en utilisant `Docker Compose`.

VISA 3

4.4. Version 4: PHP & MariaDB & API Python & Authentification JWT

- Modifiez l'API Python avec `FastAPi` qui permet l'authentification et la délivrance de Jeton JWT
- Modifiez votre programme `PHP` pour affiner les sessions déjà en place en fonction du jeton `JWT`.
- Utilisez `GitLab` pour piloter le dépôt de vos codes sources.
- Déployez l'application avec `Docker` en utilisant `Docker Compose`.

VISA 4

Pour la suite au choix, faire la version 5a ou 5b

4.5. Version 5a: PHP-MVC & MariaDB & API Python & Authentification JWT

En première année nous n'avions pas conçue l'application `PHP` en utilisant l'approche `POO` et le modèle `MVC`.

- Modifiez votre programme `PHP` afin d'utiliser l'approche `POO` et le modèle `MVC`.
- Utilisez `GitLab` pour piloter le dépôt de vos codes sources.
- Déployez l'application avec Docker en utilisant `Docker Compose`.

VISA 5a

4.6. Version 5b: Flask & MariaDB & API Python & Authentification JWT

- Suivez le tutoriel fourni pour découvrir la programmation WEB en `Python` avec `Flask`.
- Faites évoluer votre application en remplaçant les codes `PHP` par un codage en `Python` avec `Flask`.
- Utilisez `GitLab` pour piloter le dépôt de vos codes sources.
- Déployez l'application avec `Docker` en utilisant `Docker Compose`.

VISA 5b

5. Travail à rendre

Voici ci-dessous la liste des travaux à rendre :

5.1. Vous devrez déposer après chaque séance en autonomie dans votre espace de travail collaboratif (cf Slack):

- Point d'avancement séance n°XX
- Lieu de travail : Salle XX
- Tâches prévues pendant la séance :
- Tâches traitées pendant la séance :
- Problèmes rencontrés :
- Tâches prévues avant la prochaine séance :

> **Vous poserez vos questions exclusivement sous Slack (pas de mail !) dans votre canal privé.**

5.2. Calendrier des versions

1. Version 1: Semaine 40
2. Version 2: Semaine 42
3. Version 3: Semaine 49
4. Version 4: Semaine 51
5. Version 5: Semaine 06

5.3. Une archive intermediaire de votre projet **bien rangée** à chaque version

A rendre en fin de semaine un rapport de projet au format `.md` décrivant la version, les scripts Docker de déploiement et les tests.

5.4. Une archive finale de votre projet **bien rangée**

A rendre le jour de la soutenance un rapport de projet au format `.md` décrivant toutes les versions, les scripts Docker de déploiement et les tests.

5.5. Soutenance

Lors de la dernière séance, vous ferez une démonstration de votre travail.

Vous êtes incitez à prévoir une vidéo de capture de démonstration afin d'anticiper des problèmes techniques...