TP – Nouvelles technologies de la répartition 2020-21

M1 TNSI FI

L'objectif du TP est de fournir une "application" bancaire mettant en œuvre l'architecture spécifiée ci-dessous, en utilisant l'ensemble des technologies vues en cours.

La banque dispose de deux serveurs distincts. Un serveur Wildfly (10.x minimum) et un serveur Tomcat (8.5.x minimum).

Sur le serveur Wildfly, la banque héberge des services REST JSON permettant à un client utilisant une application (créée par exemple par un éditeur tiers) de consulter le solde et les opérations réalisées sur son compte. Pour simplifier, nous pouvons admettre qu'un client = un compte bancaire. Créez des jeux de tests "en dur" pour simuler la présence de clients dans une base de données.

Ce serveur expose également un ensemble de services REST XML permettant de réaliser des opérations sur les comptes bancaires des clients (crédit ou débit d'un certain montant pour le client A par exemple)

Le second serveur, Tomcat héberge une autre application (Payment gateway) avec des services utilisés par exemple par des sites web de commerce pour payer les commandes des clients. Ces services sont accessibles en SOAP et demandent le débit ou le remboursement sur des comptes de clients. Quand une demande est réalisée, ce serveur utilise les services XML du serveur Wildfly pour réaliser les opérations bancaires. On choisit de ne pas se préoccuper de la sécurité/confidentialité des échanges entre les serveurs.

La solution proposée doit pouvoir se coupler avec un site de commerce qui doit être développé par vous. Le site du commerce doit être une autre application. En total, il faudra avoir 3 apps (Banque, Gateway et Site e-commerce).

Vous pouvez déployer le site web dans : (A votre choix, mais il faudra indiquer le choix dans le rapport)

a) les instance existants du Tomcat/Wildly (c'est à dire, dans la même instance du Wildfly (e.g., localhost:8080) il aura la Banque et l'e-commerce), ou

b) nouvelles instances du même server e.g. Wildfly avec la banque port 8080, et Wildly avec l'e-commerce port 8090.

c) un autre serveur (e.g. Glassfish).

Vous proposerez, en plus des applications hébergées sur les serveurs, trois programmes permettant de tester les accès SOAP, JSON et XML proposés par les différentes applications. Vous devez également écrire des tests JUnit pour tester les différents accès.

> **TP à rendre de manière individuelle pour le Vendredi 30 Avril, 19h45 en ZIP (obligatoire) sur Moodle (cours NTP : Nouvelles Technologies de la Répartition) avec en rapport d'activité en PDF qui présente et explique la solution proposée. Le rapport doit inclure des diagrammes d' architecture (exemple diagrammes UML). Attention : le rapport ne doit pas inclure de code source.**

Aussi, il faut préparer une vidéo de 2 à 5 minutes qui montre l'utilisation du système. La vidéo doit être mise dans la plateforme YouTube et le lien au vidéo doit être mis dans le rapport.

Suggestion : Mettre le code source du système développé sur GitHub et inclure le lien dans le rapport.

Le code source du système développé doit être mis sur GitHub et doit être documenté (JavaDoc). Inclure le lien dans le rapport.

Les séances de TP serviront à poser des questions et à montrer l'avancement du projet.

Questions : matias.martinez@uphf.fr