Cahier des Charges – Projet SOA

Équipe : Ilef Dimassi ,Hazar Ghodhbani,Aya Zariat

Enseignants responsables : Ghada Feki

Enseignante de TP : Dorra Kechrid

Date : 11/12/2025

1. Introduction

1.1 Contexte

Le projet vise à développer un système universitaire basé sur une **architecture SOA**, permettant la gestion des utilisateurs, des étudiants et des cours. Chaque fonctionnalité sera exposée sous forme de **services web RESTful ou SOAP** pour assurer l'interopérabilité entre systèmes.

1.2 Objectifs

- Fournir un système modulaire et interopérable.
- Développer des services sécurisés pour la gestion des utilisateurs et des données académiques.
- Assurer un accès simple et centralisé via une **API Gateway**.

2. Périmètre du projet

2.1 Services inclus

1. **Service Authentification** : gestion des utilisateurs et des tokens JWT.
2. **Service Étudiants** : CRUD des informations des étudiants.
3. **Service Cours** : gestion des cours et emplois du temps.

3. Besoins fonctionnels

3.1 Service Authentification

- **Type :** REST
- **Technologie :** Spring Boot
- **Fonctionnalités :**
    - Création et gestion des comptes utilisateurs (admin, professeur, étudiant)
    - Authentification via **login/password**
    - Génération et validation de **JWT** pour sécuriser l'accès aux autres services
    - Réinitialisation de mot de passe
- **Contraintes non fonctionnelles :**
    - Sécurisation avec HTTPS
    - Temps de réponse < 1 seconde

3.2 Service Étudiants

- **Type :** REST
- **Technologie :** Node.js/Express
- **Fonctionnalités :**
  - CRUD des étudiants (Créer, Lire, Mettre à jour, Supprimer)
  - Recherche d'étudiants par nom, ID ou filière
  - Liaison avec le service Auth pour la vérification des droits
- **Contraintes non fonctionnelles :**
  - Interopérabilité avec les autres services
  - Réponses en JSON
  - Conteneurisation via Docker

3.3 Service Cours

- **Type :** SOAP
- **Technologie :** Java/JAX-WS
- **Fonctionnalités :**
  - CRUD des cours et emplois du temps
  - Association des étudiants aux cours
  - Consultation des cours disponibles par semestre et par filière
- **Contraintes non fonctionnelles :**
  - Communication sécurisée via WS-Security
  - Format XML standardisé
  - Compatible avec d'autres services REST

4. Besoins non fonctionnels

Sécurité

- Authentification obligatoire via JWT pour les services REST.
- WS-Security pour le service SOAP.
- Mots de passe hashés.

Interopérabilité

- REST (JSON) ↔ SOAP (XML) via API Gateway.
- Services indépendants et conteneurisés (Docker).

Performance

- Temps de réponse < 1 seconde.
- Haute disponibilité (redémarrage via Docker).

Documentation

- Swagger pour services REST.
- WSDL auto-généré pour SOAP.

4. Contraintes techniques

- **Service Auth** : Spring Boot, Spring Security, JWT
- **Service Étudiants** : Node.js, Express, MongoDB/MySQL
- **Service Cours** : JAX-WS, Java 8+, XML
- **Conteneurisation** : Docker + docker-compose
- **Routage** : Spring Cloud API Gateway

7. Architecture globale