TP 3: Sécurisation d'une Application E-Learning avec OAuth2 / OpenID Connect (Keycloak + React + Spring Boot)

Objectif général

Mettre en place une authentification moderne basée sur **OAuth2 + OIDC** dans une architecture composée :

- d'un **serveur d'identité : Keycloak**
- d'un **backend API : Spring Boot**
- d'un **frontend SPA : React**

L'objectif est de créer une plateforme E-Learning sécurisée où :

- **STUDENT** peut consulter les cours
- **ADMIN** peut gérer les cours

Contexte du projet

L'université déploie une nouvelle plateforme e-learning.
La sécurité, la gestion centralisée des utilisateurs et le Single Sign-On sont des exigences essentielles.

Votre mission consiste à :

- Configurer Keycloak
- Sécuriser Spring Boot avec OAuth2 Resource Server
- Intégrer React avec OIDC via keycloak-js
- Gérer les rôles et les accès aux différentes sections de l'application

PARTIE 1 — Configuration du serveur d'identité Keycloak

1. Installer Keycloak et créer un **Realm : elearning-realm**.
2. Créer un **Client : react-client**
    - Type : *Public*
    - Activer : **Standard Flow (OIDC)**
    - Redirect URI : http://localhost:3000/*
3. Créer deux rôles :
    - **ROLE_ADMIN**
    - **ROLE_STUDENT**

4. Créer deux utilisateurs :
    - user1 → **ROLE_STUDENT**
    - admin1 → **ROLE_ADMIN**
5. Vérifier que l'endpoint :
   …/protocol/openid-connect/userinfo
   renvoie bien les informations du profil connecté.

PARTIE 2 — Sécurisation du backend Spring Boot

1. Créer un projet Spring Boot avec les dépendances :
    - Spring Web
    - Spring Security
    - OAuth2 Resource Server
2. Activer la validation JWT via l'issuer Keycloak.
3. Créer les endpoints :
    - GET /courses → accessible à STUDENT et ADMIN
    - POST /courses → réservé à ADMIN
    - GET /me → renvoie les claims du token
4. Activer la sécurité par rôle avec **@PreAuthorize**.
5. Tester les appels API via Postman :
    - obtenir un token **STUDENT** → vérifier accès interdit au POST
    - obtenir un token **ADMIN** → vérifier accès autorisé

PARTIE 3 — Intégration du frontend React avec Keycloak

1. Créer une app React (CRA ou Vite).
2. Installer **keycloak-js**.
3. Initialiser OIDC :
    - Authentification automatique au démarrage
    - Gestion du token et refresh
4. Récupérer les informations utilisateur via l'endpoint **/userinfo**.
5. Récupérer les rôles depuis le backend via **/me**.
6. Protéger l'interface React :
    - Section "Cours disponibles" → STUDENT + ADMIN

- Section “Gestion des cours” → ADMIN
- Afficher le prénom, nom, email, etc.

7. Ajouter un bouton **Logout** (redirection Keycloak).

PARTIE 4 — Communication sécurisée React → Spring Boot

1. Dans chaque appel API, envoyer le token :
2. Authorization: Bearer <access_token>
3. Récupérer depuis React :
   - Liste des cours (GET /courses)
   - Ajouter un cours (POST /courses) → ADMIN uniquement
4. Gérer les erreurs :
   - 401 → token invalide
   - 403 → rôle insuffisant
5. Mettre en place une redirection vers Keycloak en cas d’expiration de session.

PARTIE 5 — Livrable attendu est un rapport contenant :

1. **Schéma d’architecture** (Keycloak ↔ React ↔ Spring Boot).
2. **Captures d’écran** :
   - Connexion réussie
   - Informations du profil
   - Rôles affichés dans React
   - Appels API autorisés / interdits