COMPLIANCE ASSESSMENT

• Description générale de l'architecture

L'architecture définie pour l'application est organisée autour de deux grandes couches :
- Un **Front-End** en **Angular 19.2.15** sous forme de **SPA**
- Un **Back-End** en **Spring 6.2.8** sous forme d'**API REST** et d'**API WebSocket**

Le déploiement de ces couches se fera sur une **infrastructure Cloud conteneurisée** via **Docker**.

Le choix de cette structure vise à favoriser la **modularité**, la **maintenabilité**, mais aussi une **intégration fluide avec les systèmes internes existants** via des **interfaces standardisées**, et le modèle de déploiement choisi permet, quand à lui, de garantir la **scalabilité** et la **résilience** de l'application.

La sécurité est assurée par **une authentification JWT**, un **contrôle d'accès par rôles**, et des **échanges sécurisés via HTTPS**. Le respect du **RGPD** est intégré à la conception : **seules les données nécessaires sont conservées**, et les données des utilisateurs sont supprimées à leur demande.

Une **stratégie de déploiement en deux phases** a été adoptée afin de **prioriser les fonctionnalités critiques** (réservation) et de **différer les fonctionnalités de messagerie en temps réel** (WebSocket).

• Liste de contrôle de l'architecture terminée

Composants logiciels

• L'**architecture en couche** est respectée
  • Le **front-end** et le **back-end** sont séparés
  • Le back-end dispose d'une **couche contrôleur**
  • Le back-end dispose d'une **couche service**
  • Le back-end dispose d'une **couche repository**
  • Les différentes couches du back-end se font passer les données sous forme de **DTO**

• Les contrôleurs chargés des réservations et de la consultation de données sont des **APIs REST documentées (via Swagger)**
• Les contrôleurs chargés des interactions en temps réel sont des **APIs WebSocket**
• La **séparation des responsabilités** entre les composants est claire et respectée

Services ou composants tiers logiciels

• Le back-end est en **Spring 6.2.8**
• Le front-end est en **Angular 19.2.15**
• Les **bibliothèques tierces** utilisées doivent être **à jour** et **maintenues**
• Les **APIs tierces** doivent être **accessibles via REST**, et des **mécanismes de gestion d'erreurs et timeout** doivent être mis en place

Gestion des données

• La **base de données relationnelle** doit être **modélisée selon le schéma fourni**
• L'accès aux données doit être sécurisé (via **Spring Data JPA**)
• Le **RGPD** doit être respecté (minimum de données stockées, suppression à la demande, consentement de l'utilisateur…)

Infrastructure

• Les composants doivent être **conteneurisés via Docker**
• Le déploiement doit être fait sur une **plateforme Cloud.** Ici, on choisira **AWS.**
• Le déploiement doit être automatisé via **CI/CD**

Sécurité

• Authentification via **JWT** avec **expiration au bout d'un mois**
• Autorisation basée sur les **rôles**
• Communications client/serveur sécurisées via **HTTPS**
• Sécurisation des points d'entrée des APIs via la vérification des permissions et la validation des entrées