

## Formation Spring Intermédiaire : Web Services REST + Persistance avec Spring Data JPA

<b>Durée :</b>	5 jours
<b>Public :</b>	Développeurs Java EE
<b>Pré-requis :</b>	Maîtriser la programmation orientée objet en Java - Notions de SQL
<b>Objectifs :</b>	Maîtriser l'utilisation de Spring Boot, Web et Security pour la construction de web services REST - Implémenter une couche performance de persistance de données dans une base relationnelle
<b>Sanction :</b>	Attestation de fin de stage mentionnant le résultat des acquis
<b>Taux de retour à l'emploi:</b>	Aucune donnée disponible
<b>Référence:</b>	JAV101965-F
<b>Note de satisfaction des participants:</b>	4,72 / 5

### Comprendre l'architecture

- Architecture en couches : du monolythe au microservices
- Contraintes d'architecture des microservices
- Gestion de l'authentification centralisée dans une architecture microservices
- Intérêt d'une passerelle d'API
- Gestion centralisée des traces

### Développement de web services avec Spring Boot

- Spring Boot : principe, fonctionnalités, pré-requis
- Configuration du projet (.properties ou .yml) et utilisation de profils ou d'une configuration externe
- Configuration de Logback pour la gestion des logs (logback.xml)
- Organisation des couches du projet : controllers, services, repositories
- Intérêt d'une couche de DTOs, utilisation d'un mapper d'objets
- Implémentation de contrôleurs REST : mapping global ou spécifique, méthodes, types de retours, annotations jackson
- Gestion des paramètres de méthodes et du mapping
- Gestion du download
- Gestion de l'upload, configuration
- Gestion des services et des transactions associées
- Gestion du cross origin et restriction des domaines appelants
- Test de l'api REST avec Postman
- Ecriture de tâches asynchrones, planification
- Cache web

## **Atelier : Écriture de micro-services avec Spring web - Test des méthodes de services avec Postman ou autre**

### **Documenter une API REST**

Open API Specification (Swagger) : présentation, outil  
Utilisation de Spring Doc Open API UI  
Visualisation avec Swagger Editor  
Documentation du code Java, génération de javadoc

### **Atelier : Documentation de l'api**

### **Intercepter des requêtes et gérer les erreurs**

ControllerAdvice et gestion globale des exceptions  
Capture d'exceptions personnalisées (@ExceptionHandler)  
Intercepteurs de requêtes/réponses

### **Atelier : Gestion des exceptions et implémentation d'intercepteurs**

### **Appeler d'autres API REST (écriture de clients)**

RestTemplate : présentation, méthodes  
Ecriture de requêtes GET, POST, PUT, DELETE - utilisation de la méthode exchange()  
Gestion des paramètres et du corps de la requête  
Gestion des headers  
Gestion des réponses et utilisation d'object mappers

### **Atelier : Implémentation de clients Java pour un service REST**

### **Sécuriser un service web**

Gestion des données d'entête  
Gestion de la sécurité avec Spring Security  
Gestion des utilisateurs et des rôles

### **Atelier : Intégration de Spring Security**

### **Tester une application Spring Boot**

Stratégies de tests, types supportés  
Configuration de l'application  
Mocking des couches de l'application  
Tests auto-configurés  
Exécution et reporting

### **Atelier : implémentation et exécution de tests**

### **Configurer un projet Spring Boot pour intégrer Spring Data JPA**

Spring Data JPA : Présentation, fonctionnalités, dépendances Maven  
Configuration d'un projet Spring Boot  
Propriétés par défaut et paramétrage  
Gestion des logs avec Logback

## **Atelier : Intégration de Spring Data JPA dans un projet, configuration des traces**

### **Réaliser le mapping des entités et des opérations**

Mapping des tables et gestion des clés primaires (simples, composées)  
Mapping des types de bases, propriétés des colonnes  
Gestion de la concurrence : optimistic (versioning), pessimistic  
Gestion des relations : OneToMany/ManyToOne, OneToOne, ManyToMany  
Paramétrage des cascades  
Gestion des collections : Map, Set, List, ...  
Mapping de l'héritage  
Stratégies de chargement : Lazy ou Eager

### **Atelier : Réalisation d'un schéma global de mapping d'une base de données, opérations CRUD (Create Read Update Delete)**

### **Ecrire des requêtes JP-QL ou SQL**

Interface JpaRepository et ses dérivées, ancêtres : méthodes disponibles  
Nommage de méthodes pour une auto-génération des requêtes  
Requêtes JPQL ou natives avec @Query : jointures, paramètres, fetch  
Repository personnalisé et injection de l'EntityManager  
Gestion des procédures stockées

### **Atelier : Ecriture de repositories et test depuis des services ou des contrôleurs**

### **Maîtriser des concepts avancées**

Cache : fonctionnement, niveaux  
Configuration du cache : @Cacheable  
Mise en place d'une solution d'audit de tables (historique de modifications)

### **Atelier : Implémentation d'une couche complète de persistance - mise en place d'un cache**