OSD
OPEN DIGITAL SERVICE

Scannez-moi

WHATSAPP
+213 674 05 38 95

SITE WEB
osdigitalis.com

FICHE FORMATION · OSD

Développer des API avec Java Spring Boot

Concevoir, sécuriser, tester et déployer une application back-end Java professionnelle

30 heures | 10 séances | projet fil rouge | API REST

Une formation pratique pour **construire une API REST complète avec Spring Boot**, de l'initialisation du projet jusqu'au déploiement en conteneur, avec les pratiques des équipes de développement professionnelles.

30 h	10 × 3 h	intermédiaire	projet
DURÉE	SÉANCES	NIVEAU	FORMAT

Description

Spring Boot est le cadre de référence pour les applications Java côté serveur : banques, administrations, plateformes de commerce et services en ligne l'utilisent pour leurs API et leurs microservices. Maîtriser Spring Boot, c'est accéder à une large part des offres d'emploi en développement back-end.

Tout au long de la formation, chaque participant construit **la même application de bout en bout** : une API de gestion (catalogue, commandes, utilisateurs) avec base de données, sécurité, tests et documentation. Chaque séance ajoute une couche, jusqu'à une application prête à être déployée avec **Docker**.

Public visé

- Étudiants en informatique ayant suivi un cours de Java
- Développeurs PHP, Python ou .NET qui veulent passer à Spring
- Développeurs Java qui découvrent Spring Boot

Prérequis

- Bases de Java et de la programmation orientée objet (classes, héritage, interfaces)
- Notions de SQL (SELECT, INSERT, jointures)
- Un PC avec 8 Go de RAM, JDK 21 et un IDE (IntelliJ IDEA ou VS Code)

À l'issue de la formation, vous saurez

☐	structurer un projet **Spring Boot** avec Maven ou Gradle et comprendre l'**injection de dépendances** ;
☐	exposer une **API REST** propre : contrôleurs, DTO, validation et gestion des erreurs ;
☐	persister des données avec **Spring Data JPA** et gérer les relations et les migrations ;
☐	sécuriser l'API avec **Spring Security** et des jetons **JWT** ;
☐	écrire des **tests** unitaires et d'intégration (JUnit, Mockito) ;
☐	documenter l'API avec **OpenAPI** et la déployer dans un conteneur **Docker**.

Méthode pédagogique

- Courts exposés, puis **codage en direct** et ateliers à chaque séance.
- Un **projet fil rouge** qui grandit de séance en séance, versionné avec Git.
- Bonnes pratiques professionnelles : architecture en couches, revue de code, tests, configuration par environnement.
- Exercices progressifs avec corrections commentées.

Programme — 10 séances

1 **Écosystème Java et premier projet Spring Boot** 3 h

Place de Spring dans l'écosystème Java, Spring Initializr, structure d'un projet Maven, démarrage de l'application et premier point d'accès HTTP.

`Spring Initializr` `Maven` `JDK 21`

2 **Injection de dépendances et configuration** 3 h

Conteneur Spring, beans, composants et services, injection par constructeur, fichiers de configuration et profils par environnement.

`IoC` `@Service` `profils`

3 **Construire une API REST** 3 h

Contrôleurs REST, verbes HTTP, codes de retour, DTO et mapping, tests manuels des points d'accès avec un client HTTP.

`@RestController` `DTO` `Postman`

4 **Persistance avec Spring Data JPA** 3 h

Entités, dépôts, requêtes dérivées et personnalisées, connexion à PostgreSQL ou MySQL, migrations de schéma avec Flyway.

`JPA` `PostgreSQL` `Flyway`

5 **Relations et requêtes avancées** 3 h

Relations un-à-plusieurs et plusieurs-à-plusieurs, chargement paresseux, pagination, tri et filtres, problème N+1 et ses solutions.

`relations` `pagination` `N+1`

6 **Validation et gestion des erreurs** 3 h

Validation des entrées, messages d'erreur cohérents, gestionnaire global d'exceptions, format d'erreur standard pour les clients de l'API.

`Bean Validation` `@ControllerAdvice` `ProblemDetail`

7 **Sécurité : authentification et autorisation** 3 h

Spring Security, chiffrement des mots de passe, authentification par jeton JWT, rôles et protection des points d'accès.

`Spring Security` `JWT` `rôles`

8 **Tests automatisés** 3 h

Tests unitaires avec JUnit et Mockito, tests d'intégration des contrôleurs et de la base, couverture et organisation des tests.

`JUnit 5` `Mockito` `MockMvc`

9 **Documentation, observabilité et Docker** 3 h

Documentation OpenAPI, journalisation, Actuator et métriques, création d'une image Docker et lancement avec Docker Compose.

`OpenAPI` `Actuator` `Docker`

10 **Projet final et soutenance** 3 h

Finalisation du projet fil rouge, revue de code croisée, démonstration de l'API complète et soutenance individuelle.

`projet` `revue de code` `soutenance /20`

Évaluation et validation

- Ateliers vérifiés à chaque séance, projet versionné sur Git.
- **Projet final** : API complète, testée et documentée, présentée en soutenance et notée sur 20.
- Attestation de formation délivrée par OSD — Open Digital Service.

Modalités

Durée	30 heures, en 10 séances de 3 heures
Format	présentiel en salle informatique, ou intra-entreprise / établissement
Effectif	groupes de 8 à 20 participants
Matériel	un poste par participant avec JDK 21, IntelliJ IDEA ou VS Code, Docker et un client HTTP
Supports remis	fiches de séance, énoncés d'ateliers, code source de référence du projet
Formateur	Mohammed Slimani — développeur fullstack senior et architecte DevOps, plus de 20 ans d'expérience

Intéressé(e) par cette formation ?

Programme adaptable (microservices, Spring Cloud, volume horaire, public). Contact : contact@osdigitalis.com · WhatsApp +213 674 05 38 95 · osdigitalis.com

Open Digital Service

Mohammed Slimani — Formateur & consultant

+213 (0)6 74 05 38 95 · contact@osdigitalis.com · msslimani@osdigitalis.com

osdigitalis.com · LinkedIn : in/mohammed-slimani-osdigitalis