Secure Java Development Essentials (OSCC-SJD) Syllabus

Course Summary, Methodology, and Organization of Content	
Course Summary	SJD-100: Secure Java Development Essentials is a course designed to equip Java developers with essential security practices to protect software from common vulnerabilities. You'll learn secure coding principles, error handling, input validation, output encoding, and session management techniques critical for developing robust Java applications. Through comprehensive instruction and hands-on labs, you'll gain practical experience implementing security at every stage of the development process. SJD-100 prepares you for the Java-developer-specific OffSec CyberCore (OSCC-SJD) certification exam, enabling you to demonstrate your expertise in secure Java development practices.
Learning Methodology	SJD-100 utilizes a blended learning approach that combines interactive online instruction with hands-on labs. Learners engage with comprehensive course materials, including readings and practical exercises, while applying their knowledge in simulated environments to develop practical skills. Completing the course and successfully passing the associated exam awards the OffSec CyberCore Secure Java Development (OSCC-SJD) certification.

The following section contains the various Learning Modules and Learning Units.

Learning Module	Learning Units
Introduction to CyberCore: Secure Java Development Essentials	Intended Audience
	Course Structure
	How to Succeed
	Getting Help

	Virtual Machines Overview
	The SJD-100: CyberCore - Secure Java Development Essentials Exam
	Wrapping Up
Secure Coding Principles with Java	The CIA Triad
	Authentication and Authorization
	Handling Input and Output
	Least Privilege
	Defense in Depth
	Failing Safely
	Common Coding Principles and Misconceptions
	Case Study: Insecure Direct Object Reference
	Wrapping Up

Error Handling and Logging with Java	Error Handling in Java
	Logging and Monitoring
	Case Study: Detecting an Attack
	Wrapping Up
Input Validation with Java	Approaching Input Validation
	Input Validation to Prevent Attacks
	Case Study: Server-Side Request Forgery
	Wrapping Up
Output Encoding with Java	Introduction to Output Encoding
	HTML Entity Encoding
	URL Encoding
	Automatic Encoding with Template Engines
	Case Study: Cross-site Scripting

	Wrapping Up
HTTP Cookie Security with Java	HTTP Cookie Overview
	HTTP Cookie Attributes for Security
	Case Study: Cross-site Request Forgery
	Wrapping Up
Security Misconfigurations with Java	Risks Unrelated to Code
	Reducing Risks with Security Configurations
	Introduction to Content Security Policy
	Case Study: Mitigating Cross-Site Scripting With CSP
	Wrapping Up
Web Session Management with Java	Authenticating Users
	Case Study: Proper Credential Storage and Preventing Brute Force Attacks

	Wrapping Up
Using Databases with Java	Interacting with Databases Using Java
	Database Hardening
	Handling Sensitive Data in Databases
	Case Study: SQL Injection
Assembling the Pieces: Java Security Essentials	Introduction to SJD Challenge Labs
	Application Overview
	Threat Modeling
	Security Reports
	Putting It All Together: Capstone Exercises
	Wrapping Up