Java Level 2 Training

Day 1 – Java Recap & Object-Oriented Programming

Objective: Refresh fundamentals and strengthen OOP for independent coding.

Topics:

1. **Java Basics Recap**
    - Variables (primitive vs reference types)
    - Operators (arithmetic, relational, logical, bitwise)
    - Control Flow (if-else, switch, loops)
    - Methods (parameters, return types, overloading)
2. **OOP Essentials**
    - Classes & Objects, Constructors
    - Inheritance (extends keyword, method overriding)
    - Polymorphism (compile-time vs runtime)
    - Abstraction (abstract classes, methods)
    - Encapsulation (access modifiers, getters/setters)
3. **Advanced OOP**
    - Interfaces & multiple inheritance
    - Inner Classes (static, member, anonymous)
    - Composition vs Inheritance (HAS-A vs IS-A relationship)

Handon Labs:

- **Bank Account System**: Implement deposits, withdrawals, and balance check using classes & methods.
- **Employee Class Hierarchy**: Create base class Employee and subclasses (Manager, Developer) demonstrating inheritance & polymorphism.
- **Debugging in IDE**: Set breakpoints, inspect variables, step into/step over methods.

Day 2 – Java APIs, Exceptions & Multithreading

Objective: Build robust and concurrent programs.

Topics:

1. **Collections Framework**
    - List (ArrayList, LinkedList)
    - Set (HashSet)
    - Map (HashMap)
    - Queue (PriorityQueue)
2. **Exception Handling**
    - Checked vs Unchecked exceptions
    - try-catch-finally block
    - Throwing & creating Custom Exceptions

    - Best practices (specific exceptions, avoiding empty catch blocks)
3. **Functional Programming**
    - Lambdas: `(args) -> expression`
    - Streams API: map, filter, reduce, collect
4. **Multithreading Basics**
    - Creating threads: `Thread` vs `Runnable`
    - Thread lifecycle & Synchronization

Handon Labs:

- **Library Management System**: Store books in a `List`, search/filter using Streams API.
- **Custom Exception**: Create `InvalidBookException` for invalid data input.
- **Multi-threaded Ticket Booking System**: Multiple threads booking seats.
- **Employee Records with Stream API**: Filter employees by salary, department, etc.

Day 3 – Git, Build Tools & JUnit Testing

Objective: Learn enterprise development tools for collaboration & testing.

Topics:

1. **Git Essentials**
    - Git init, clone, add, commit, push, pull
    - Branching & merging
    - Handling merge conflicts
2. **Build Tools**
    - **Maven**: POM.xml structure, dependencies, lifecycle (compile, test, package, install)
    - **Gradle**: build.gradle scripts, dependency management
    - Maven vs Gradle comparison
3. **JUnit Testing**
    - Lifecycle: @BeforeEach, @AfterEach, @BeforeAll, @AfterAll
    - Assertions: `assertEquals`, `assertTrue`, `assertThrows`
    - Parameterized tests
    - Basics of TDD (red → green → refactor cycle)

Handon Labs:

- **Git Repo Setup**: Initialize repo, create branches, push to GitHub.
- **Maven Project Creation**: Create & run simple HelloWorld app.
- **Convert Maven → Gradle Project**: Add Gradle build scripts.
- **JUnit Tests for Services**: Write unit tests for Calculator or Employee Service class.

Day 4 – Spring Boot Foundations

Objective: Build REST APIs with Spring Boot.

Topics:

1. **Spring Core Concepts**
    - Inversion of Control (IoC), Dependency Injection (DI)
    - Bean lifecycle, @Component, @Autowired
2. **Spring Boot Basics**
    - Starters (spring-boot-starter-web, spring-boot-starter-data-jpa)
    - Auto-configuration
    - Project structure & application.properties
3. **REST Controllers**
    - @RestController, @RequestMapping
    - GET, POST, PUT, DELETE endpoints
    - RequestBody vs PathVariable vs RequestParam
4. **Service Layer**
    - Separation of concerns
    - Business logic in @Service classes

Handon Labs:

- **Create Spring Boot Project**: Using Spring Initializr.
- **Employee Service**: CRUD operations with hardcoded data.
- **Build REST APIs**: Expose endpoints for employee CRUD.
- **Test with Postman**: Call APIs and check responses.

Day 5 – JPA, Spring Testing & Guided Mini Project

Objective: Persist data with JPA and integrate testing.

Topics:

1. **Spring Data JPA**
    - Entities, @Id, @GeneratedValue
    - Repositories (CrudRepository, JpaRepository)
    - ORM basics & Hibernate overview
    - Custom Queries with @Query
2. **Database Integration**
    - H2 Database setup (in-memory)
    - Schema auto-generation
3. **Spring Testing**
    - @SpringBootTest for integration testing
    - Repository & service layer testing

Lab (Guided Mini Project):

Employee Management System – Phase 1

- Create Entity: Employee (id, name, department, salary)
- CRUD APIs with JPA repository
- Unit tests for Employee repository & service

- Push project to GitHub

Day 6 – Validation, Exception Handling & Optimizing REST APIs

Objective: Apply validation, centralized exception handling and optimize REST APIs.

Topics:

- Global exception handling using @RestControllerAdvice
- Custom exception handling
- Request validation using @Valid & Hibernate Validator (e.g., @NotNull, @Email, @Size)
- Pagination & filtering for large datasets (`Pageable`)
- Caching with Spring Cache

Lab (Enhance Mini Project):

- Add validation to Employee entity (name not empty, salary > 0)
- Implement GlobalExceptionHandler with @ControllerAdvice
- Add pagination API for employee list
- Add caching for "Get All Employees" API

Day 7 – Spring Security & Monitoring with Actuator

Objective: Implement spring security and monitoring with actuator.

Topics:

- Securing REST APIs with Spring Security
- Enabling Spring Boot Actuator endpoints
- Health checks & metrics (health, info, metrics)
- Securing actuator endpoints

Lab (Enhance Mini Project):

- Add Spring Security to Employee Management System
- Secure endpoints with Basic Auth
- Enable actuator endpoints & check health, metrics
- Restrict actuator endpoints

Day 8 – Sessions, JWT & OWASP Essentials

Objective: Secure Spring Boot apps using sessions, cookies, JWT, and OWASP.

Topics:

- Session management & cookies in Spring Boot
- JWT Authentication flow (login → token → verify → access)
- Role-based authorization for APIs
- OWASP Top 10 Security Practices

Lab (Secure Mini Project):

- Implement JWT Authentication (login → token generation)
- Apply role-based access to Employee APIs
- Add secure headers
- Apply OWASP best practices

Day 9 & 10 – Capstone Project & Presentations

Objective: Apply all concepts to a full-fledged system.

Project Ideas:

1. **Online Course Management System** – Manage courses, students, enrolments
2. **Inventory Management System** – Track products, stock, suppliers
3. **Online Task Manager** – CRUD + multithreading for scheduling tasks
4. **Employee Payroll System** – Calculate salaries, maintain payroll history

Requirements:

- Spring Boot backend with REST APIs
- JPA persistence with DB
- GitHub repo for collaboration
- JUnit tests for at least 3 critical services
- Request validation & exception handling
- JWT authentication + role-based access
- Apply OWASP Top 10 best practices

Deliverables:

- Working Spring Boot application
- Database persistence with JPA
- Unit test coverage
- Code hosted on GitHub
- Final demo & architecture explanation