# Java Technologies - Lab 3

**[valid 2025-2026]**

**Dependency Injection and Cross-Cutting Concerns**

**Compulsory** (1p)

- Create a simple, illustrative scenario that demonstrates all major types of dependency injection points in Spring Boot.
- Your example must also prove the order in which injection is performed: constructor → field → setter & method.

---

**Homework** (2p)

Create a Spring Boot Order Management System that handles Customers, Orders, and Discounts. Different clients (companies using your system) require different discount policies:

- Loyal customers get a percentage discount;
- Orders with a large value get a fixed amount discount;
- No discount (default).

Design a pluggable discount mechanism where Spring selects the appropriate DiscountService implementation depending on the client.

- Define the domain model: Customer, Order, DiscountService and a simple in-memory repository of customers.
- Create the implementations for the discount policies.
- Use Spring Profiles or a custom configuration property to decide which discount strategy to inject.
- Create the OrderService which applies discounts using the injected DiscountService.
- Every time a discount is applied, log the method name, customer name, and discount amount.
- Create a decorator (using aspects) that, before applying a discount, checks if the customer exists and is eligible to get a discount. If not, throw a custom exception.
- Publish an event whenever a discount exceeds a certain amount and implement a listeners for that type of event.

---

**Advanced** (2p)

- Consider the case when you have decision-heavy logic that could grow or change frequently.
  Use a rule-engine, such as Drools (or define your own) to keep your code clean, maintainable, and easier to adapt to new business requirements.
  Implement the discount policies using the rule-engine.
- Create a reusable Spring Boot Starter that auto-configures an auditing aspect.