**Configuring Beans with Annotations**

**Enabling Component Scanning (Step 1):**

* This part shows the need of adding the context namespace declaration to applicationContext.xml. This allows you to use context-related elements like <context:component-scan>.
* The <context:component-scan base-package="com.library" /> element instructs Spring to scan classes within the com.library package and its sub-packages for components annotated with stereotypes like @Service and @Repository.

**Annotating Classes (Step 2):**

* Annotations provide a cleaner and more concise way to define beans.
  + @Service: Annotates the BookService class, marking it as a Spring service component.
  + @Repository: Annotates the BookRepository class, marking it as a Spring data access repository component.

**Autowiring Dependencies (Step 2):**

* The @Autowired annotation on the setter method of BookService injects the BookRepository dependency automatically when Spring creates a BookService instance.
* This eliminates the need for explicit configuration of the setter in XML.

**Testing the Configuration (Step 3):**

* Running LibraryManagementApplication confirms that Spring successfully discovers and manages the beans based on annotations.

**Summary:**

* Migration from XML configuration to a more maintainable approach using annotations has done.
* Spring automatically discovers and injects dependencies based on component scanning and annotations.

**Extra Notes:**

* Annotations offer a cleaner separation of concerns compared to XML configuration.
* This approach promotes code reusability and reduces boilerplate code.