**LAPORAN HASIL PRAKTIKUM 4**

**PEMROGRAMAN BERORIENTASI OBJEK LANJUT**

**“Spring-3”**

**Dosen Pengampu :**

**Sri Hartati Wijono, M.Kom.**



Oleh

Nama : Maria Gresia Plena Br Purba

NIM : 235314094

Kelas : DP

**PROGRAM STUDI INFORMATIKA**

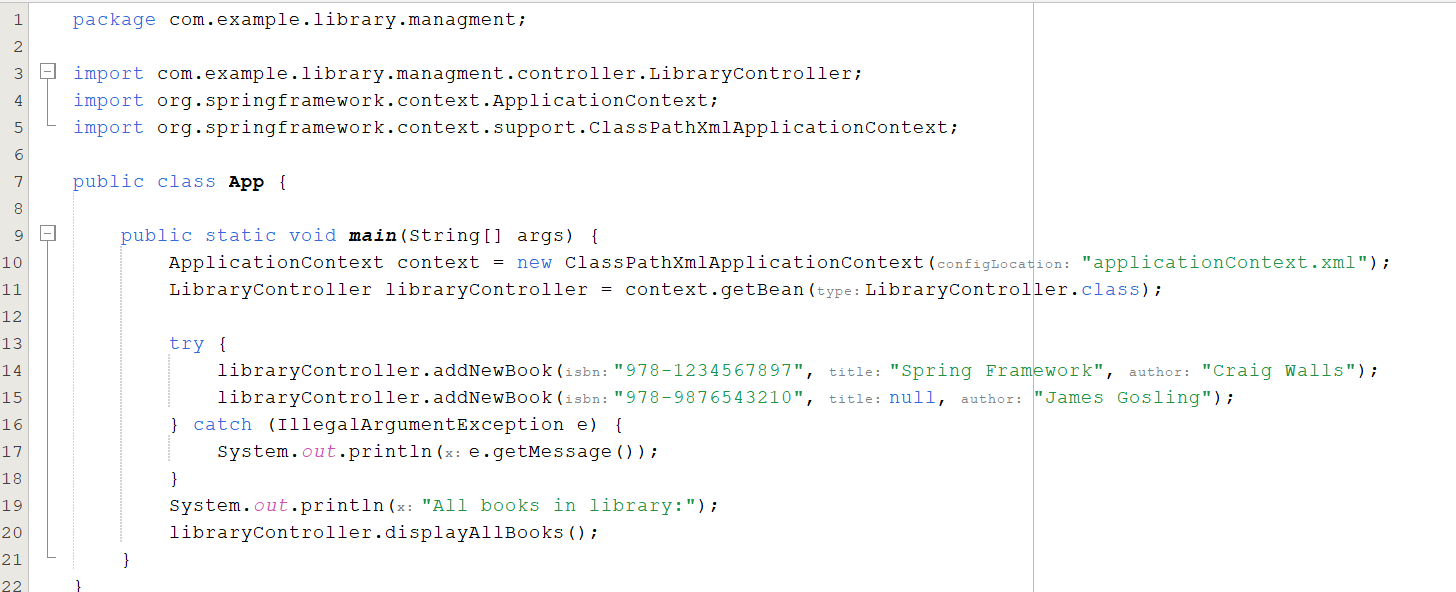
**FAKULTAS SAINS DAN TEKNOLOGI**

**UNIVERSITAS SANATA DHARMA**

**YOGYAKARTA**

**2024**

1. **TUJUAN**
2. Memahami dan menerapkan **Spring AOP** untuk memisahkan logika lintas aplikasi (cross-cutting concerns) seperti logging, keamanan, dan validasi.
3. Menerapkan **Spring Transaction Management** untuk mengelola transaksi di aplikasi.
4. **PRAKTIKUM**
   1. **Capture code class App**

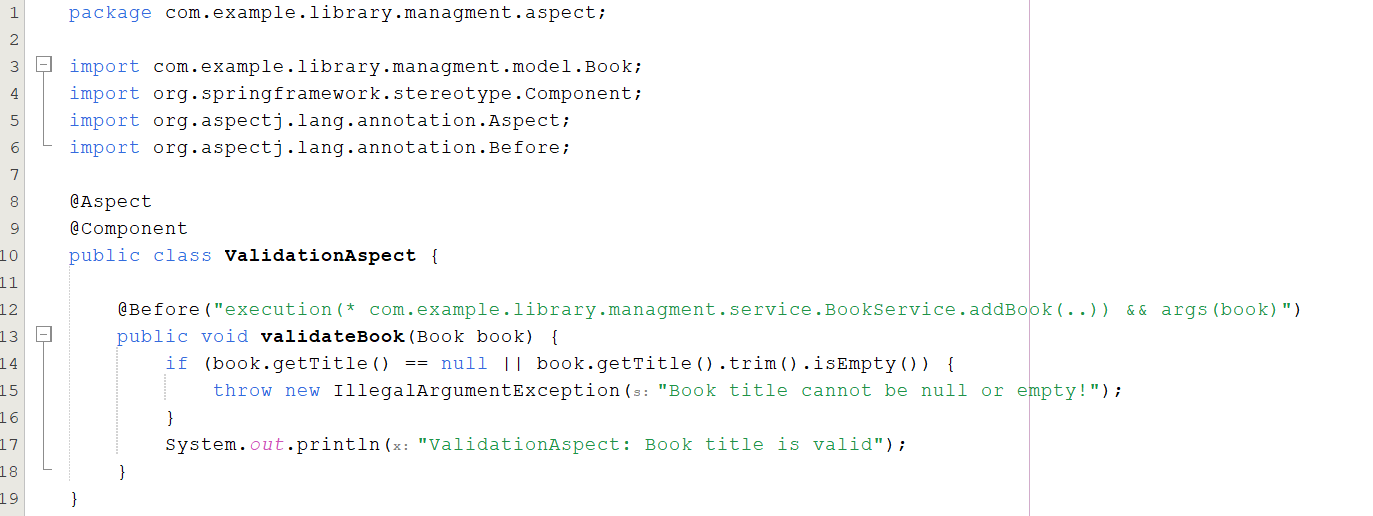
**Penjelasan :** class ini mengimplementasikan penggunaan dari ApplicationContext yang digunakan untuk mengkonfigurasi “ApplicationContext.xml” dengan menggunakan ClassPathXmlApplicationContext. Lalu, menggunakan getBean() untuk mengambil bean LibraryController.class. Menggunakan try and catch untuk menangani pengecualian agar program tidak langsung berhenti jika terjadi kesalahan. Kemudian, memanggil method addNewBook() untuk menambahkan buku baru dan menampilkan daftar buku dengan memanggil method displayAllBooks().

* 1. **Capture code class LoggingAspect**

****

**Penjelasan : @Aspect** untuk mendefinisikan bahwa class ini merupakan Spring Bean yang dikelola oleh Spring IoC Container. **@Component** berarti class akan didaftarkan oleh Spring sebagai bean yang dikelola oleh IoC Container. **@Before** berarti menjalankan advice sebelum eksekusi method BookService. Method logBeforeServiceMethods() berfungsi untuk menampilkan pesan jika suatu method di dalam BookService telah dipanggil.

* 1. **Capture code class ValidationAspect**

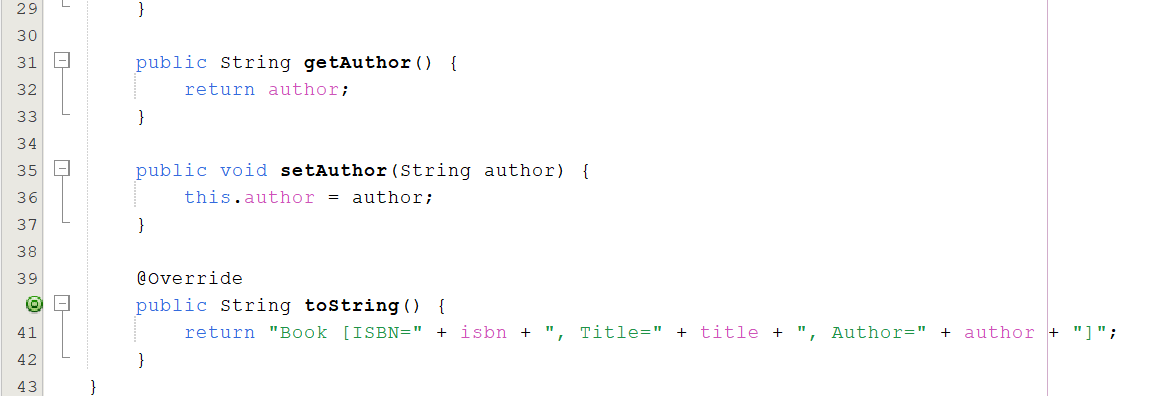
**Penjelasan : @Aspect** untuk mendefinisikan bahwa class ini merupakan Spring Bean yang dikelola oleh Spring IoC Container. **@Component** berarti class akan didaftarkan oleh Spring sebagai bean yang dikelola oleh IoC Container. **@Before** berarti menjalankan advice sebelum eksekusi method addBook. Mehod validateBook berfungsi untuk mengetahui apakah judul buku sudah valid atau tidak. Jika tidak valid, maka akan menampilkan pesan “ValidationAspect: Book title is valid”.

* 1. **Capture code class LibraryController**



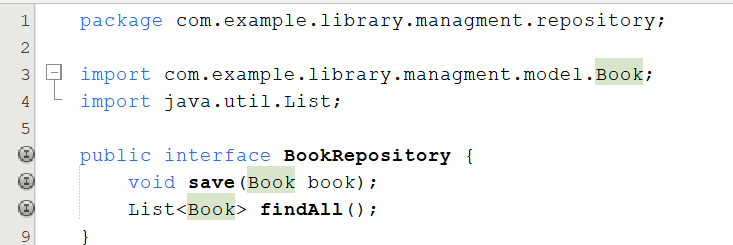
**Penjelasan : @Controller** berfungsi untuk membuat class ini sebagai Spring MVC Controller. **@Autowired** berfungsi untuk melakukan Dependency Injection atau tidak. Method addNewBook() berfungsi untuk membuat objek pada class Book dengan nama book yang memiliki parameter berisi isbn, title, author. Setelah objek Book dibuat, method ini akan memanggil method addBook() dengan parameter book yang akan disimpan ke dalam objek bookService untuk menambahkan buku baru. Method displayBooks() berfungsi untuk mengembalikan dann menampilkan daftar buku. Pada method ini, terdapat pemanggilan bookService.getAllBooks() yang berfungsi untuk mendapatkan buku-buku yang ada. Lalu, akan melakukan perulangan untuk mencetak informasi dari buku.

* 1. **Capture code class Book**

****

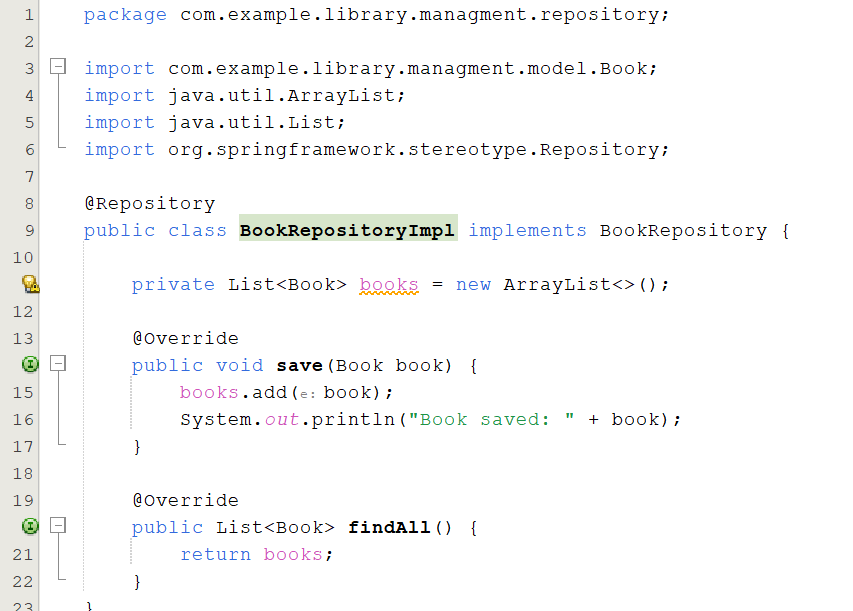
**Penjelasan :** terdapat beberapa method dengan setter dan getter. Getter berfungsi untuk mengembalikan nilai dari variabel-variabel, sedangkan setter untuk mengubah nilai dari variabel. Kemudian, terdapat method toString() yang akan mengembalikan informasi dari buku berupa isbn, title, dan author yang telah disimpan ke dalam variabel yang ada.

* 1. **Capture code interface BookRepository**

****

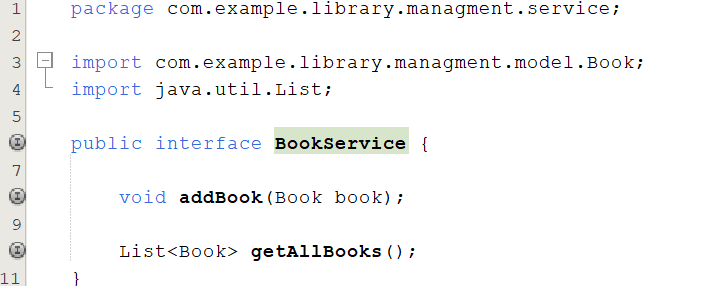
**Penjelasan :** interface ini memiliki method save() dengan isian parameter book yang bertipe Book. Method ini berfungsi untuk menyimpan ataupun menambahkan buku baru ke dalam objek Book. Lalu, terdapat method findAll() yang berfungsi untuk mengembalikan daftar dalam bentuk List<Book>.

* 1. **Capture code class BookRepositoryImpl**

****

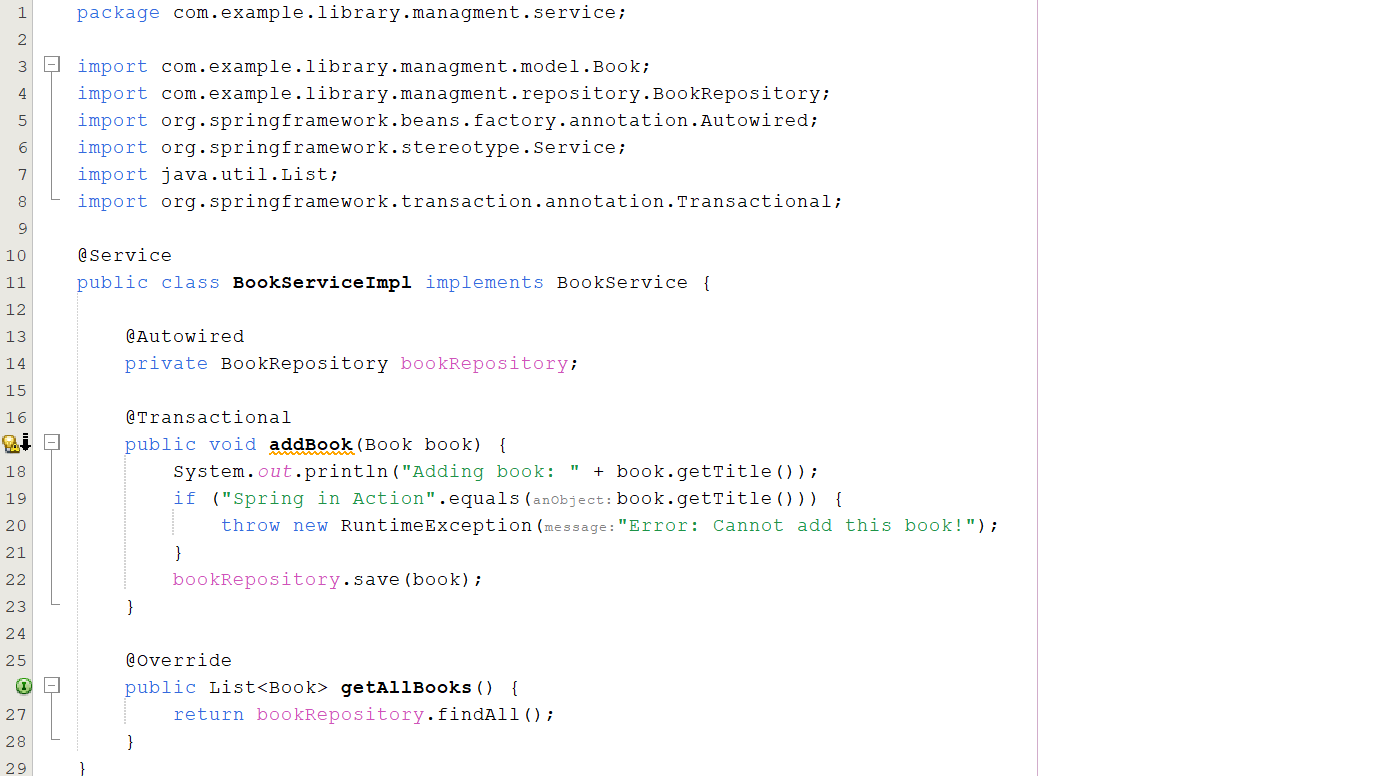
**Penjelasan :** class ini mengimplementasikan interface BookRepository. Pada class ini terdapat variabel books yang bertipe List<books> yang digunakan untuk menyimpan daftar buku dalam bentuk implementasi ArrayList<>. Kemudian, terdapat method save() dengan parameter book yang bertipe Book. Method ini akan memanggil method add pada objek books dengan parameter berisi book dan mencetak pesan jika buku tersimpan (“Book saved”). Method findAll() akan mengambil dan mengembalikan buku-buku yang ada setelah disimpan ke dalam objek Book.

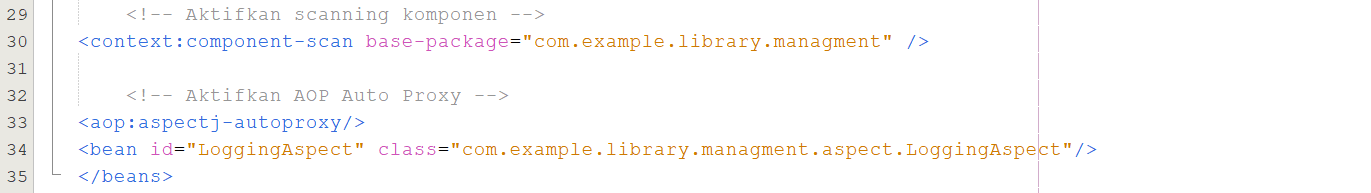
* 1. **Capture code interface BookService**

****

**Penjelasan :** interface ini memiliki method addBook() dengan parameter book yang bertipe Book. Method ini berfungsi untuk menambahkan objek Book baru. Method getAllBooks() akan mengambil dan mengembalikan buku-buku yang ada setelah disimpan ke dalam objek Book dalam bentuk List<Book>.

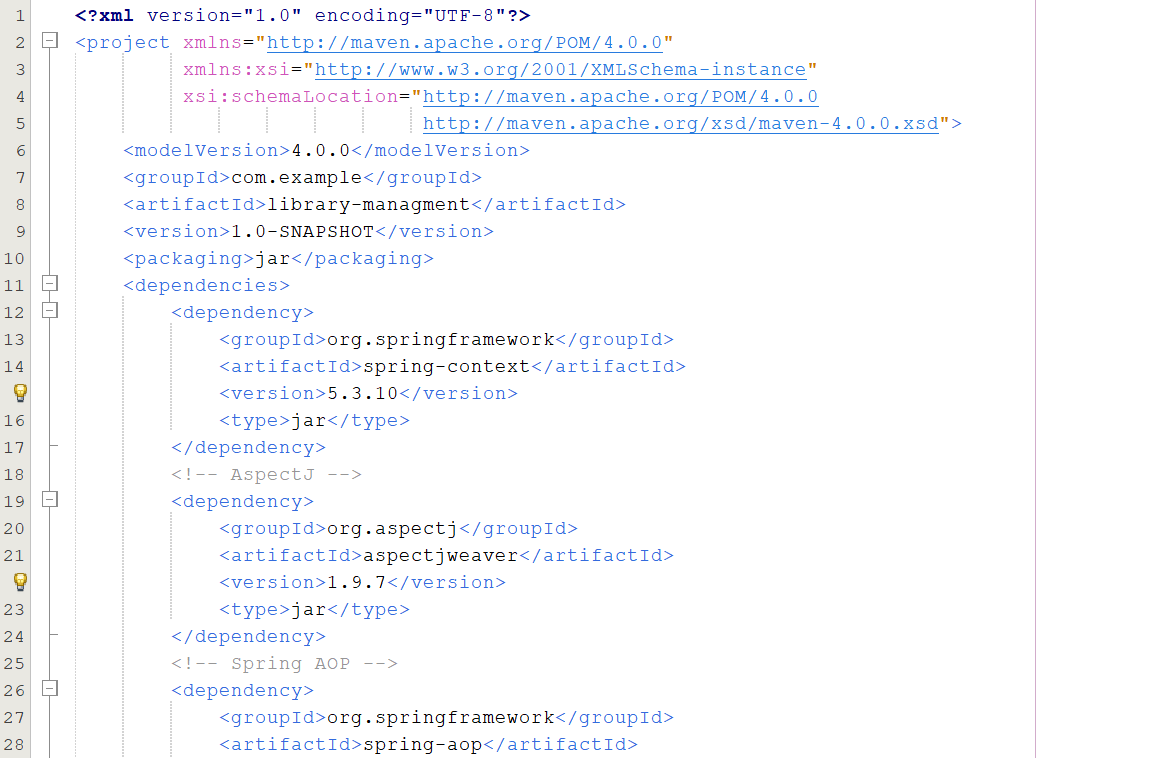
* 1. **Capture code class BookServiceImpl**

**Penjelasan : @Service** berfungsi untuk memproses logika bisnis. **@Autowired** berfungsi untuk melakukan Dependency Injection atau tidak. **@Transactional** untuk mengelola transaksi secara otomatis. Method addBook() berfungsi untuk menambahkan buku dengan memanggil method getTitle() pada objek book. Jika judul buku yang dicari tidak sama dengan “Spring in action”, maka akan menampilkan pesan. Lalu, memanggil mehod save() pada bookRepository. Method getAllBooks() akan mengambil dan mengembalikan buku-buku yang ada setelah disimpan ke dalam objek Book dalam bentuk List<Book>.

* 1. **ApplicationContext.xml**

**Penjelasan :** mengatur h2 database (DriverManagerDataSource), Transaction Manager (DataSourceTransactionManager), mengaktifkan scanning komponen pada com.example.library.managment, dan mengaktifkan AOP dengan <aop:aspectj-autoproxy/>.

* 1. **Pom.xml**

**Penjelasan :** pada bagian dependencies terdapat spring-context yang digunakan untuk mendukung fitur dari Spring, aspectjweaver untuk mengaktifkan fitur AOP menggunakan AspectJ, spring-aop untuk mendukung AOP dalam Spring, spring-tx untuk mendukung transaksi, H2 database, dan spring-jdbc mendukung operasi database.

1. **DAFTAR PUSTAKA**

**-**