**Exercise 10: Online Bookstore - Configuring Content Negotiation**

**1. Configure Spring Boot for Content Negotiation**

By default, Spring Boot supports both JSON and XML for content negotiation. However, you can explicitly configure content negotiation using a ContentNegotiationConfigurer.

**Add Dependencies for XML Support**

<dependency>

<groupId>com.fasterxml.jackson.dataformat</groupId>

<artifactId>jackson-dataformat-xml</artifactId>

</dependency>

Configure Content Negotiation in WebMvcConfig

@Configuration

public class WebMvcConfig implements WebMvcConfigurer {

@Override

public void configureContentNegotiation(ContentNegotiationConfigurer configurer) {

// Enable content negotiation using the Accept header and default to JSON

configurer.favorParameter(false)

.ignoreAcceptHeader(false)

.defaultContentType(MediaType.APPLICATION\_JSON)

.mediaType("json", MediaType.APPLICATION\_JSON)

.mediaType("xml", MediaType.APPLICATION\_XML);

}

}

2. **Produce and Consume Different Media Types Based on the Accept Header**

@RestController

@RequestMapping("/api/books")

public class BookController {

@Autowired

private BookService bookService;

// Get a book by ID, producing JSON or XML based on the Accept header

@GetMapping(value = "/{id}", produces = { MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })

public ResponseEntity<Book> getBookById(@PathVariable Long id) {

Book book = bookService.getBookById(id);

return ResponseEntity.ok(book);

}

// Get all books, producing JSON or XML based on the Accept header

@GetMapping(produces = { MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })

public ResponseEntity<List<Book>> getAllBooks() {

List<Book> books = bookService.getAllBooks();

return ResponseEntity.ok(books);

}

}

3. **Handling Requests with Different Content Types**

@PostMapping(consumes = { MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE },

produces = { MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })

public ResponseEntity<Book> createBook(@RequestBody Book book) {

Book createdBook = bookService.saveBook(book);

return ResponseEntity.status(HttpStatus.CREATED).body(createdBook);

}

4. **Testing Content Negotiation**