Case Study 1: Hospital Management System (XML-Based Configuration)

Java Code:

1. Patient.java

```
package com.example.hospital;

public class Patient {
    public void registerPatient() {
        System.out.println("Patient registered successfully.");
    }

    public void getPatientDetails() {
        System.out.println("Displaying patient details...");
    }
}
```

2. Appointment.java

```
package com.example.hospital;

public class Appointment {
    public void bookAppointment() {
        System.out.println("Appointment booked.");
    }

    public void cancelAppointment() {
        System.out.println("Appointment cancelled.");
    }
}
```

3. Billing.java

```
package com.example.hospital;

public class Billing {
    public void generateBill() {
        System.out.println("Generating bill...");
    }

    public void sendBill() {
        System.out.println("Bill sent via email.");
    }
}
```

4. HospitalService.java

```
package com.example.hospital;

public class HospitalService {
    private Patient patient;
    private Appointment appointment;
    private Billing billing;

    // Setters for injection
    public void setPatient(Patient patient) {
        this.patient = patient;
    }

    public void setAppointment(Appointment appointment) {
        this.appointment = appointment;
    }
```

```
    public void setBilling(Billing billing) {
        this.billing = billing;
    }

    public void performOperations() {
        patient.registerPatient();
        appointment.bookAppointment();
        billing.generateBill();
        billing.sendBill();
    }
}
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
       http://www.springframework.org/schema/beans/spring-beans.xsd">

    <bean id="patient" class="com.example.hospital.Patient" />
    <bean id="appointment" class="com.example.hospital.Appointment" />
    <bean id="billing" class="com.example.hospital.Billing" />

    <bean id="hospitalService" class="com.example.hospital.HospitalService">
        <property name="patient" ref="patient"/>
        <property name="appointment" ref="appointment"/>
        <property name="billing" ref="billing"/>
```

```
  </bean>
</beans>
```

MainApp.java

```
package com.example.hospital;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MainApp {
   public static void main(String[] args) {
      ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
      HospitalService service = (HospitalService) context.getBean("hospitalService");
      service.performOperations();
   }
}
```

.

Case Study 2: E-Commerce Order Processing (Java-Based Configuration)

1. Product.java

```
package com.example.ecommerce;

public class Product {
   public void addProduct() {
      System.out.println("Product added to inventory.");
   }
```

```
    public void listProducts() {
        System.out.println("Listing all products...");
    }
}
```

2. Order.java

```
package com.example.ecommerce;

public class Order {
    public void createOrder() {
        System.out.println("Order created successfully.");
    }

    public void cancelOrder() {
        System.out.println("Order has been cancelled.");
    }
}
```

3. Payment.java

```
package com.example.ecommerce;

public class Payment {
    public void processPayment() {
        System.out.println("Payment processed successfully.");
    }

    public void refundPayment() {
        System.out.println("Payment has been refunded.");
    }
}
```

4. EcommerceService.java

```
package com.example.ecommerce;

public class EcommerceService {
    private Product product;
    private Order order;
    private Payment payment;

    public EcommerceService(Product product, Order order, Payment payment) {
        this.product = product;
        this.order = order;
        this.payment = payment;
    }

    public void executeEcommerceOperations() {
        product.addProduct();
        product.listProducts();
        order.createOrder();
        payment.processPayment();
    }
}
```

AppConfig.java

```
package com.example.ecommerce;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
```

```
@Configuration
public class AppConfig {

    @Bean
    public Product product() {
        return new Product();
    }

    @Bean
    public Order order() {
        return new Order();
    }

    @Bean
    public Payment payment() {
        return new Payment();
    }

    @Bean
    public EcommerceService ecommerceService() {
        return new EcommerceService(product(), order(), payment());
    }
}
```

MainApp.java

```
package com.example.ecommerce;
```

```
import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class MainApp {
   public static void main(String[] args) {
      ApplicationContext context = new
AnnotationConfigApplicationContext(AppConfig.class);
      EcommerceService service = context.getBean(EcommerceService.class);
      service.executeEcommerceOperations();
   }
}
```

Case Study 3: Library Management System (Annotation-Based Configuration)

Java Classes

1. Book.java

```
package com.example.library;

import org.springframework.stereotype.Component;

@Component
public class Book {
   public void addBook() {
```

```
        System.out.println("Book added to library.");
    }

    public void searchBook() {
        System.out.println("Searching for books...");
    }
}
```

---

2. Member.java

```
package com.example.library;

import org.springframework.stereotype.Component;

@Component
public class Member {
    public void registerMember() {
        System.out.println("New member registered.");
    }

    public void viewMembers() {
        System.out.println("Displaying all members...");
    }
}
```

---

3. Loan.java

```
package com.example.library;
```

```
import org.springframework.stereotype.Component;

@Component
public class Loan {
    public void issueBook() {
        System.out.println("Book issued to member.");
    }

    public void returnBook() {
        System.out.println("Book returned to library.");
    }
}
```

---

4. LibraryService.java

```
package com.example.library;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class LibraryService {

    @Autowired
    private Book book;

    @Autowired
    private Member member;
```

```
    @Autowired
    private Loan loan;

    public void operateLibrary() {
        book.addBook();
        member.registerMember();
        loan.issueBook();
        loan.returnBook();
    }
}
```

---

5. MainApp.java

```
package com.example.library;

import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

@Configuration
@ComponentScan("com.example.library")
public class MainApp {
    public static void main(String[] args) {
        ApplicationContext context = new
AnnotationConfigApplicationContext(MainApp.class);
        LibraryService service = context.getBean(LibraryService.class);
        service.operateLibrary();
```

```
    }
}
```