Day 8 _Spring_Daywise_Assignment

Case Study 1: XML-Based Configuration
Case Study Title: Hospital Management System

Patient.java

```java
package com.example.hospital;

public class Patient {
    private String name = "Harsha";
    private int age = 23;
    private String gender = "Male";

    public void registerPatient() {
        System.out.println("Patient Registered Successfully:");
        System.out.println("Name: " + name + ", Age: " + age + ", Gender: " + gender);
    }

    public void getPatientDetails() {
        System.out.println("Fetching Patient Details...");
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Gender: " + gender);
    }
}
```

Appointment.java

```java
package com.example.hospital;

public class Appointment {
    private String doctor = "Dr. Akash";
    private String date = "2025-08-01";
    private String time = "10:30 AM";

    public void bookAppointment() {
        System.out.println("Booking Appointment...");
        System.out.println("Doctor: " + doctor);
        System.out.println("Date: " + date);
        System.out.println("Time: " + time);
    }

    public void cancelAppointment() {
        System.out.println("Appointment with " + doctor + " on " + date + " at " + time + " is cancelled.");
    }
}
```

Billing.java

```java
package com.example.hospital;

public class Billing {
    private double consultationFee = 500.0;
    private double labCharges = 300.0;

    public void generateBill() {
        double total = consultationFee + labCharges;
        System.out.println("Generating Bill...");
        System.out.println("Consultation Fee: ₹" + consultationFee);
        System.out.println("Lab Charges: ₹" + labCharges);
        System.out.println("Total Bill: ₹" + total);
    }

    public void sendBill() {
        System.out.println("Bill has been emailed to harsha@example.com");
    }
}
```

HospitalService.java

```java
package com.example.hospital;

public class HospitalService {

    private Patient patient;
    private Appointment appointment;
    private Billing billing;

    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 manageHospital() {
        patient.registerPatient();
        patient.getPatientDetails();

        appointment.bookAppointment();
        appointment.cancelAppointment();

        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
                           https://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>
```

Main Class

```
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.manageHospital();  // Executes all features
    }
}
```

pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" ...>
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.example</groupId>
    <artifactId>hospital-management-xml</artifactId>
    <version>1.0</version>

    <dependencies>
        <!-- Spring Core -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.32</version>
        </dependency>
    </dependencies>
</project>
```

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

Product.java

```
package com.example.ecommerce;

import java.util.ArrayList;
import java.util.List;

public class Product {
    private List<String> products = new ArrayList<>();

    public void addProduct(String productName) {
        products.add(productName);
        System.out.println("Product added: " + productName);
    }

    public void listProducts() {
        System.out.println("Available Products:");
        for (String p : products) {
            System.out.println("- " + p);
        }
    }
}
```

Order.java

```
package com.example.ecommerce;

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

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

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 refunded successfully.");
    }
}
```

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 runEcommerceFlow() {
        product.addProduct("Laptop");
        product.addProduct("Smartphone");
        product.listProducts();

        order.createOrder();
        payment.processPayment();

        order.cancelOrder();
        payment.refundPayment();
    }
}
```

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());
    }
}
```

<u>Main Method</u>

```
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.runEcommerceFlow();
    }
}
```

<u>pom.xml</u>

```
<project xmlns="http://maven.apache.org/POM/4.0.0" ...>
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.example</groupId>
    <artifactId>ecommerce-java-config</artifactId>
    <version>1.0</version>

    <dependencies>
        <!-- Spring Core -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.32</version>
        </dependency>
    </dependencies>
</project>
```

Case Study 3: Annotation-Based Configuration
Case Study Title: Library Management System

pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" ...>
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.example</groupId>
    <artifactId>library-annotation-config</artifactId>
    <version>1.0</version>

    <dependencies>
        <!-- Spring Core Context -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.32</version>
        </dependency>
    </dependencies>
</project>
```

Book.java

```
package com.example.library;

import org.springframework.stereotype.Component;

@Component
public class Book {

    public void addBook() {
        System.out.println("Book added: 'Python'");
    }

    public void searchBook() {
        System.out.println("Searching for 'Python'... Book found.");
    }
}
```

Member.java

```
package com.example.library;

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

```
@Component
public class Member {

    public void registerMember() {
        System.out.println("Member registered: Harsha Mukartihal");
    }

    public void viewMembers() {
        System.out.println("Viewing all members... Harsha Mukartihal is active.");
    }
}
```

Loan.java

```
package com.example.library;

import org.springframework.stereotype.Component;

@Component
public class Loan {

    public void issueBook() {
        System.out.println("Book 'Python' issued to Harsha.");
    }

    public void returnBook() {
        System.out.println("Book 'Python' returned by Harsha.");
    }
}
```

LibraryService.java

```
package com.example.library;

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

@Component
public class LibraryService {

    @Autowired
    private Book book;

    @Autowired
    private Member member;

    @Autowired
    private Loan loan;

    public void manageLibrary() {
        book.addBook();
        book.searchBook();

        member.registerMember();
        member.viewMembers();

        loan.issueBook();
        loan.returnBook();
    }
}
```

<u>MainApp.java</u>

```
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.manageLibrary();
    }
}
```