Case Study Title: Online Course Enrollment System

Scenario:

An educational startup wants to build a basic web application for students to view available courses and enroll online. The company has a small IT team familiar with Java and wants to use Spring MVC to ensure the application follows a clean, maintainable structure based on MVC architecture.

//Course.java

```
public class Course {
   private int id;
   private String name;
   private String description;
   public Course() {}
         public Course(int id, String name, String description) {
                  this.id = id;
                  this.name = name;
                  this.description = description;
         }
         public int getId() {
                  return id;
         }
         public void setId(int id) {
                  this.id = id;
         }
         public String getName() {
                  return name;
         }
         public void setName(String name) {
                  this.name = name;
         }
         public String getDescription() {
            return description;
        }
        public void setDescription(String description) {
            this.description = description;
        }
}
```

//Student.java

```
package com.example.course.model;
public class Student {
  private String name;
  private String email;
  private int courseId;
        public Student(String name, String email, int courseId) {
            this.name = name;
            this.email = email;
            this.courseId = courseId;
        }
        public Student() {}
        public String getName() {
            return name;
        }
        public void setName(String name) {
            this.name = name;
        }
        public String getEmail() {
            return email;
        }
        public void setEmail(String email) {
            this.email = email;
        }
        public int getCourseId() {
                return courseId;
        }
        public void setCourseId(int courseId) {
                this.courseId = courseId;
        }
}
```

//CourseController

```
package com.example.course.controller;
import com.example.course.model.Course;
import com.example.course.model.Student;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import java.util.*;
@Controller
public class CourseController {
  private static List<Course> courses = new ArrayList<>();
  static {
    courses.add(new Course(1, "Java Basics", "Learn Java from scratch."));
    courses.add(new Course(2, "Spring MVC", "Intro to Spring MVC framework."));
    courses.add(new Course(3, "Web Development", "HTML, CSS, JavaScript overview."));
  }
  @GetMapping("/courses")
  public String showCourses(Model model) {
    model.addAttribute("courses", courses);
    return "courses";
  }
  @GetMapping("/enroll")
  public String showEnrollmentForm(@RequestParam("id") int courseId, Model model) {
    Student student = new Student();
    student.setCourseId(courseId);
```

```
        model.addAttribute("student", student);
        return "enroll";
    }
    @PostMapping("/submitEnrollment")
    public String submitEnrollment(@ModelAttribute Student student, Model model) {
        model.addAttribute("student", student);
        String courseName = courses.stream()
            .filter(c -> c.getId() == student.getCourseId())
            .map(Course::getName)
            .findFirst().orElse("Unknown");
        model.addAttribute("courseName", courseName);
        return "success";
    }
}
```

//courses.jsp

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html><body>
    <h2>Available Courses</h2>
    <ul>
        <c:forEach var="course" items="${courses}">
            <li>
                <strong>${course.name}</strong> - ${course.description}
                <a href="enroll?id=${course.id}">Enroll</a>
            </li>
        </c:forEach>
    </ul>
</body></html>
```

//enroll.jsp

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html><body>
    <h2>Enrollment Form</h2>
```

```
  <form action="submitEnrollment" method="post">
    <input type="hidden" name="courseId" value="${student.courseId}" />
    Name: <input type="text" name="name" required /><br/>
    Email: <input type="email" name="email" required /><br/>
    <input type="submit" value="Enroll" />
  </form>
</body></html>
```

//success.jsp

```
<html>
<head><title>Success</title></head>
<body>
  <h2>Enrollment Successful</h2>
  <p>Thank you, ${student.name}!</p>
  <p>You have enrolled in: ${courseName}</p>
</body>
</html>
```

//web.xml

```
<web-app xmlns="http://jakarta.ee/xml/ns/jakartaee"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://jakarta.ee/xml/ns/jakartaee
     http://jakarta.ee/xml/ns/jakartaee/web-app_5_0.xsd"
     version="5.0">
  <servlet>
    <servlet-name>dispatcher</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>/WEB-INF/dispatcher-servlet.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
```

```
    <servlet-mapping>
        <servlet-name>dispatcher</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
</web-app>
```

//dispatcher.servlet

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:mvc="http://www.springframework.org/schema/mvc"
       xmlns:context="http://www.springframework.org/schema/context"
       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
                           http://www.springframework.org/schema/mvc
                           http://www.springframework.org/schema/mvc/spring-mvc.xsd
                           http://www.springframework.org/schema/context
                           http://www.springframework.org/schema/context/spring-context.xsd">

    <context:component-scan base-package="com.example.course" />
    <mvc:annotation-driven />

    <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/views/" />
        <property name="suffix" value=".jsp" />
    </bean>
</beans>
```

//pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
         http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
```

```
  <groupId>com.example</groupId>
  <artifactId>course-enrollment</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>war</packaging>
  <properties>
    <maven.compiler.source>17</maven.compiler.source>
    <maven.compiler.target>17</maven.compiler.target>
  </properties>
  <dependencies>
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-webmvc</artifactId>
      <version>5.3.33</version>
    </dependency>
    <dependency>
      <groupId>jakarta.servlet</groupId>
      <artifactId>jakarta.servlet-api</artifactId>
      <version>6.0.0</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>jstl</groupId>
      <artifactId>jstl</artifactId>
      <version>1.2</version>
    </dependency>
  </dependencies>
</project>
```

Case Study Title: Online Shopping Portal –

Order Processing Monitoring

Scenario Description:

An online shopping portal provides a service class OrderService that has three key methods:

1. addToCart(String product)

2. placeOrder(String orderId) 3. cancelOrder(String orderId) As a developer, you want to add cross-cutting concerns like:

• Logging when methods start (@Before)

• Logging after successful method execution (@AfterReturning)

• Logging errors when a method fails (@AfterThrowing)

• Performing cleanup or logging after any method execution, success or failure (@After)

```
//OrderService.java

package com.example.shop.service;

import com.example.shop.exception.OrderNotFoundException;

import org.springframework.stereotype.Component;

@Component

public class OrderService {

   public void addToCart(String product) {

      System.out.println("Product added to cart: " + product);

   }

   public void placeOrder(String orderId) {

      if ("INVALID_ID".equals(orderId)) {

         throw new OrderNotFoundException("Order ID is invalid: " + orderId);

      }

      System.out.println("Order placed successfully: " + orderId);

   }

   public void cancelOrder(String orderId) {
        System.out.println("Order cancelled: " + orderId);
    }
}
```
//OrderNotFoundException.java
```
package com.example.shop.exception;


public class OrderNotFoundException extends RuntimeException {
    public OrderNotFoundException(String message) {
        super(message);
    }
}
```
//OrderLoggingAspect.java
```
package com.example.shop.aspect;
import org.aspectj.lang.JoinPoint;
import org.aspectj.lang.annotation.*;
import org.springframework.stereotype.Component;
@Aspect
@Component
public class OrderLoggingAspect {
    @Before("execution(* com.example.shop.service.OrderService.*(..))")
    public void logBefore(JoinPoint joinPoint) {
        System.out.println("Starting method: " + joinPoint.getSignature().getName() +
                        " with arguments: " + java.util.Arrays.toString(joinPoint.getArgs()));
    }
    @AfterReturning(pointcut = "execution(* com.example.shop.service.OrderService.*(..))", returning
= "result")
    public void logAfterReturning(JoinPoint joinPoint, Object result) {
        System.out.println(" Method " + joinPoint.getSignature().getName() + " executed successfully.");
    }
    @AfterThrowing(pointcut = "execution(* com.example.shop.service.OrderService.*(..))", throwing
= "ex")
    public void logAfterThrowing(JoinPoint joinPoint, Throwable ex) {
```

```java
        System.out.println(" Exception in method: " + joinPoint.getSignature().getName() +
                " — " + ex.getClass().getSimpleName() + ": " + ex.getMessage());
    }
    @After("execution(* com.example.shop.service.OrderService.*(..))")
    public void logAfter(JoinPoint joinPoint) {
        System.out.println(" Method " + joinPoint.getSignature().getName() + " execution finished");
    }
}
```

applicationContext.xml

```xml
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:aop="http://www.springframework.org/schema/aop"
       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
         http://www.springframework.org/schema/context
         http://www.springframework.org/schema/context/spring-context.xsd
         http://www.springframework.org/schema/aop
         http://www.springframework.org/schema/aop/spring-aop.xsd">
    <!-- Component scan -->
    <context:component-scan base-package="com.example.shop" />
    <!-- Enable AOP -->
    <aop:aspectj-autoproxy />
</beans>
```

MainApp.java

```java
package com.example.shop;
import com.example.shop.service.OrderService;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
```

```java
public class MainApp {
    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
        OrderService service = context.getBean(OrderService.class);
        System.out.println("\n 1. Testing addToCart()");
        service.addToCart("Laptop");
        System.out.println("\n 2. Testing placeOrder() with valid ID");
        service.placeOrder("ORD123");
        System.out.println("\n 3. Testing placeOrder() with invalid ID");
        try {
            service.placeOrder("INVALID_ID");
        } catch (Exception e) {
            // Handled by AOP
        }
        System.out.println("\n 4. Testing cancelOrder()");
        service.cancelOrder("ORD123");
    }
}
```

//Pom.xml

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
                      http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.example</groupId>
    <artifactId>online-shopping-portal</artifactId>
    <version>1.0-SNAPSHOT</version>
    <dependencies>
        <!-- Spring Core -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.33</version>
        </dependency>
        <!-- Spring AOP -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-aspects</artifactId>
            <version>5.3.33</version>
        </dependency>
        <!-- AspectJ -->
        <dependency>
            <groupId>org.aspectj</groupId>
            <artifactId>aspectjweaver</artifactId>
            <version>1.9.21</version>
        </dependency>
    </dependencies>
</project>
```