SPRING CORE MAVEN

Superset ID : 6384831

Name : Mohana Priya N

E-mail : mohanapriya.2205056@srec.ac.in

Mandatory Questions:

1) Exercise 1: Configuring a Basic Spring Application

Solution:

//pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" ...>
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.library</groupId>
    <artifactId>LibraryManagement</artifactId>
    <version>1.0</version>
    <dependencies>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>
    </dependencies>
</project>
```

//applicationContext.xml

```
<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="bookRepository" class="com.library.repository.BookRepository"/>
    <bean id="bookService" class="com.library.service.BookService">
        <property name="bookRepository" ref="bookRepository"/>
    </bean>
```

```
</beans>
```

//BookRepository.java

```
package com.library.repository;
public class BookRepository {
    public void saveBook() {
        System.out.println("Book saved to repository.");
    }
}
```

//BookService.java

```
package com.library.service;
import com.library.repository.BookRepository;
public class BookService {
    private BookRepository bookRepository;
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }
    public void addBook() {
        System.out.println("BookService: Adding book...");
        bookRepository.saveBook();
    }
}
```

//LibraryManagementApplication.java

```
package com.library;
import com.library.service.BookService;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class LibraryManagementApplication {
    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
        BookService bookService = context.getBean("bookService", BookService.class);
```

```
        bookService.addBook();
    }
}
```

OUTPUT:

```
BookService: Adding book...
Book saved to repository.
```

2) Exercise 2: Implementing Dependency Injection

Solution:

//pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" ...>
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.library</groupId>
    <artifactId>LibraryManagement</artifactId>
    <version>1.0</version>
    <dependencies>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>
    </dependencies>
</project>
```

//applicationContext.xml

```
<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="bookRepository" class="com.library.repository.BookRepository"/>
```

```
    <bean id="bookService" class="com.library.service.BookService">
        <property name="bookRepository" ref="bookRepository"/>
    </bean>
</beans>
```

//BookRepository.java

```
package com.library.repository;
public class BookRepository {
    public void saveBook() {
        System.out.println("Book saved to repository.");
    }
}
```

//BookService.java

```
package com.library.service;
import com.library.repository.BookRepository;
public class BookService {
    private BookRepository bookRepository;
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }
    public void addBook() {
        System.out.println("BookService: Adding book...");
        bookRepository.saveBook();
    }}
```

//LibraryManagementApplication.java

```
package com.library;
import com.library.service.BookService;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class LibraryManagementApplication {
    public static void main(String[] args) {
```

```
        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
        BookService bookService = context.getBean("bookService", BookService.class);
        bookService.addBook();
    }
}
```

OUTPUT:

```
BookService: Adding book...
Book saved to repository.
```

3) Exercise 4: Creating and Configuring a Maven Project

Solution:

//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.library</groupId>
    <artifactId>LibraryManagement</artifactId>
    <version>1.0</version>
    <dependencies>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-aop</artifactId>
```

```
        <version>5.3.34</version>
      </dependency>
    </dependencies>
    <build>
      <plugins>
        <plugin>
          <groupId>org.apache.maven.plugins</groupId>
          <artifactId>maven-compiler-plugin</artifactId>
          <version>3.8.1</version>
          <configuration>
            <source>1.8</source>
            <target>1.8</target>
          </configuration>
        </plugin>
      </plugins>
    </build>
</project>
```

OUTPUT:

```
[INFO] BUILD SUCCESS
BookService: Adding book...
Book saved to repository.
```

Other Questions:

4) Exercise 3: Implementing Logging with Spring AOP

Solution:

//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.library</groupId>
    <artifactId>LibraryManagement</artifactId>
    <version>1.0</version>
    <dependencies>
      <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>5.3.34</version>
      </dependency>
      <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-aop</artifactId>
        <version>5.3.34</version>
      </dependency>
    </dependencies>
</project>
```

//applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       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
          https://www.springframework.org/schema/beans/spring-beans.xsd
          http://www.springframework.org/schema/aop
          https://www.springframework.org/schema/aop/spring-aop.xsd">
    <aop:aspectj-autoproxy/>
    <bean id="bookRepository" class="com.library.repository.BookRepository"/>
    <bean id="bookService" class="com.library.service.BookService">
```

```
        <property name="bookRepository" ref="bookRepository"/>
    </bean>
    <bean id="loggingAspect" class="com.library.aspect.LoggingAspect"/>
</beans>
```

// BookRepository.java

```
package com.library.repository;
public class BookRepository {
    public void saveBook() {
        System.out.println("Book saved to repository.");
    }
}
```

//BookService.java

```
package com.library.service;
import com.library.repository.BookRepository;
public class BookService {
    private BookRepository bookRepository;
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }
    public void addBook() {
        System.out.println("BookService: Adding book...");
        bookRepository.saveBook();
    }
}
```

//LoggingAspect.java

```
package com.library.aspect;
import org.aspectj.lang.ProceedingJoinPoint;
import org.aspectj.lang.annotation.*;
@Aspect
public class LoggingAspect {
```

```
    @Around("execution(* com.library.service.*.*(..))")
    public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {
        long start = System.currentTimeMillis();
        Object result = joinPoint.proceed();
        long end = System.currentTimeMillis();
        System.out.println("Execution time of " + joinPoint.getSignature().getName() + ": " +
(end - start) + "ms");
        return result;
    }
}
```

//LibraryManagementApplication.java

```
package com.library;
import com.library.service.BookService;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class LibraryManagementApplication {
    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
        BookService service = context.getBean("bookService", BookService.class);
        service.addBook();
    }
}
```

OUTPUT:

```
BookService: Adding book...
Book saved to repository.
Execution time of addBook: 1ms
```

5) Exercise 5 : Configuring the Spring IoC Container

Solution:

//pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" ...>
```

```
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.library</groupId>
    <artifactId>LibraryManagement</artifactId>
    <version>1.0</version>
    <dependencies>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>
    </dependencies>
</project>
```

//applicationContext.xml

```
<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="bookRepository" class="com.library.repository.BookRepository"/>
    <bean id="bookService" class="com.library.service.BookService">
        <property name="bookRepository" ref="bookRepository"/>
    </bean>
</beans>
```

//BookRepository.java

```
package com.library.repository;
public class BookRepository {
    public void saveBook() {
        System.out.println("Book saved to repository.");
    }
}
```

//BookService.java

```
package com.library.service;
import com.library.repository.BookRepository;
public class BookService {
    private BookRepository bookRepository;
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }
    public void addBook() {
        System.out.println("BookService: Adding book...");
        bookRepository.saveBook();
    }
}
```

//LibraryManagementApplication.java

```
package com.library;
import com.library.service.BookService;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class LibraryManagementApplication {
    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
        BookService bookService = context.getBean("bookService", BookService.class);
        bookService.addBook();
    }
}
```

OUTPUT:

```
BookService: Adding book...
Book saved to repository.
```

6) Exercise 6: Configuring Beans with Annotations

Solution:

//pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" ...>
   <modelVersion>4.0.0</modelVersion>
   <groupId>com.library</groupId>
   <artifactId>LibraryManagement</artifactId>
   <version>1.0</version>
   <dependencies>
      <dependency>
         <groupId>org.springframework</groupId>
         <artifactId>spring-context</artifactId>
         <version>5.3.34</version>
      </dependency>
   </dependencies>
</project>
```

//applicationContext.xml

```
<beans xmlns="http://www.springframework.org/schema/beans"
      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
         https://www.springframework.org/schema/beans/spring-beans.xsd
         http://www.springframework.org/schema/context
         https://www.springframework.org/schema/context/spring-context.xsd">
   <context:component-scan base-package="com.library"/>
</beans>
```

//BookRepository.java

```
package com.library.repository;
import org.springframework.stereotype.Repository;
```

```
@Repository
public class BookRepository {
   public void saveBook() {
      System.out.println("Book saved to repository.");
   }}
//BookService.java
package com.library.service;
import com.library.repository.BookRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class BookService {
   @Autowired
   private BookRepository bookRepository;
   public void addBook() {
      System.out.println("BookService: Adding book...");
      bookRepository.saveBook();
   }
}
//LibraryManagementApplication.java
package com.library;
import com.library.service.BookService;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class LibraryManagementApplication {
   public static void main(String[] args) {
      ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
      BookService bookService = context.getBean("bookService", BookService.class);
      bookService.addBook();
}}
```

OUTPUT:

```
BookService: Adding book...
Book saved to repository.
```

7) Exercise 7: Implementing Constructor and Setter Injection

Solution:

//pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" ...>
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.library</groupId>
    <artifactId>LibraryManagement</artifactId>
    <version>1.0</version>
    <dependencies>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>
    </dependencies>
</project>
```

//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="bookRepository" class="com.library.repository.BookRepository"/>
    <bean id="bookService" class="com.library.service.BookService">
        <constructor-arg ref="bookRepository"/>
```

```
        <property name="bookRepository" ref="bookRepository"/>
    </bean>
</beans>
```

//BookRepository.java

```
package com.library.repository;
public class BookRepository {
    public void saveBook() {
        System.out.println("Book saved to repository.");
    }}
```

//BookService.java

```
package com.library.service;
import com.library.repository.BookRepository;
public class BookService {
    private BookRepository bookRepository;
    public BookService(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
        System.out.println("Constructor Injection used.");
    }
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
        System.out.println("Setter Injection used.");
    }
    public void addBook() {
        System.out.println("BookService: Adding book...");
        bookRepository.saveBook();
    }}
```

//LibraryManagementApplication.java

```
package com.library;
import com.library.service.BookService;
import org.springframework.context.ApplicationContext;
```

```
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class LibraryManagementApplication {
   public static void main(String[] args) {
      ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
      BookService bookService = context.getBean("bookService", BookService.class);
      bookService.addBook();
   }}
```

OUTPUT:

```
Constructor Injection used.
Setter Injection used.
BookService: Adding book...
Book saved to repository.
```

8) Exercise 8 : Implementing Basic AOP with Spring

Solution:

//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.library</groupId>
   <artifactId>LibraryManagement</artifactId>
   <version>1.0</version>
   <dependencies>
      <dependency>
         <groupId>org.springframework</groupId>
         <artifactId>spring-context</artifactId>
         <version>5.3.34</version>
      </dependency>
```

```
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-aop</artifactId>
            <version>5.3.34</version>
        </dependency>
    </dependencies>
</project>
```

//applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       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
         https://www.springframework.org/schema/beans/spring-beans.xsd
         http://www.springframework.org/schema/aop
         https://www.springframework.org/schema/aop/spring-aop.xsd">
    <aop:aspectj-autoproxy/>
    <bean id="loggingAspect" class="com.library.aspect.LoggingAspect"/>
    <bean id="bookRepository" class="com.library.repository.BookRepository"/>
    <bean id="bookService" class="com.library.service.BookService">
        <property name="bookRepository" ref="bookRepository"/>
    </bean>
</beans>
```

// BookRepository.java

```
package com.library.repository;
public class BookRepository {
    public void saveBook() {
        System.out.println("Book saved to repository.");
    }}
```

//BookService.java

```
package com.library.service;
import com.library.repository.BookRepository;
public class BookService {
   private BookRepository bookRepository;
   public void setBookRepository(BookRepository bookRepository) {
      this.bookRepository = bookRepository;
   }
   public void addBook() {
      System.out.println("BookService: Adding book...");
      bookRepository.saveBook();
   }}
```

//LoggingAspect.java

```
package com.library.aspect;
import org.aspectj.lang.JoinPoint;
import org.aspectj.lang.annotation.*;
@Aspect
public class LoggingAspect {
   @Before("execution(* com.library.service.*.*(..))")
   public void beforeAdvice(JoinPoint joinPoint) {
      System.out.println("[LOG] Before method: " + joinPoint.getSignature().getName());
   }
   @After("execution(* com.library.service.*.*(..))")
   public void afterAdvice(JoinPoint joinPoint) {
      System.out.println("[LOG] After method: " + joinPoint.getSignature().getName());——
   }}
```

//LibraryManagementApplication.java

```
package com.library;
import com.library.service.BookService;
import org.springframework.context.ApplicationContext;
```

```
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class LibraryManagementApplication {
    public static void main(String[] args) {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        BookService service = context.getBean("bookService", BookService.class);
        service.addBook();
    }
}
```

OUTPUT:

```
Before method: addBook
BookService: Adding book...
Book saved to repository.
After method: addBook
```

9) Exercise 9 : Creating a Spring Boot Application

Solution:

//LibraryManagementApplication.java

```
package com.library;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class LibraryManagementApplication {
    public static void main(String[] args) {
        SpringApplication.run(LibraryManagementApplication.class, args);
    }
}
```

//Book.java

```
package com.library.entity;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
@Entity
```

```
public class Book {
    @Id
    private int id;
    private String title;
    public Book() {}
    public Book(int id, String title) {
        this.id = id;
        this.title = title;
    }
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }
    public String getTitle() { return title; }
    public void setTitle(String title) { this.title = title; }
}
```

//BookRepository.java

```
package com.library.repository;
import com.library.entity.Book;
import org.springframework.data.jpa.repository.JpaRepository;
public interface BookRepository extends JpaRepository<Book, Integer> {
}
```

//BookController.java

```
package com.library.controller;
import com.library.entity.Book;
import com.library.repository.BookRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import java.util.List;
@RestController
@RequestMapping("/books")
public class BookController {
```

```
    @Autowired
    private BookRepository repository;
    @PostMapping
    public Book createBook(@RequestBody Book book) {
        return repository.save(book);
    }
    @GetMapping
    public List<Book> getAllBooks() {
        return repository.findAll();
    }
}
```

//application.properties

```
spring.datasource.url=jdbc:h2:mem:librarydb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=mohana
spring.datasource.password=
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.hibernate.ddl-auto=update
spring.h2.console.enabled=true
spring.h2.console.path=/h2-console
```