Exercise 1: Configuring a Basic Spring Application

Scenario:

Your company is developing a web application for managing a library. You need to use the Spring Framework to handle the backend operations.

Steps:

1. Set Up a Spring Project:

o Create a Maven project named LibraryManagement.

o Add Spring Core dependencies in the pom.xml file.

2. Configure the Application Context:

o Create an XML configuration file named applicationContext.xml in the src/main/resources directory.

o Define beans for BookService and BookRepository in the XML file.

3. Define Service and Repository Classes:

o Create a package com.library.service and add a class BookService.

o Create a package com.library.repository and add a class BookRepository.

4. Run the Application:

o Create a main class to load the Spring context and test the configuration.

Solution:

Book repository:

```
package com.library.repository;

public class BookRepository {
    public void save() {
        System.out.println("BookRepository: Saving book to database...");
    }
}
```

Book Service:

```
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 a new book...");
      bookRepository.save();
   }
}
```

MainApp:

```
package com.library;

import com.library.service.BookService;
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");

      BookService bookService = context.getBean("bookService", BookService.class);
      bookService.addBook();
   }
}
```

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

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

</beans>
```

Output:

```
[INFO]
[INFO] -------------------< com.library:library-management >-------------------
[INFO] Building library-management 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- exec:3.1.0:java (default-cli) @ library-management ---
BookService: Adding a new book...
BookRepository: Saving book to database...
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.576 s
[INFO] Finished at: 2025-07-06T10:55:40+05:30
[INFO] ------------------------------------------------------------------------
PS D:\CTS\DN-4.0--Mugilan-Superset-ID-6410310\Week3\library-management>
```

Exercise 2: Implementing Dependency Injection

Scenario:

In the library management application, you need to manage the dependencies between the BookService and BookRepository classes using Spring's IoC and DI.

Steps:

1. Modify the XML Configuration:

o Update applicationContext.xml to wire BookRepository into BookService.

2. Update the BookService Class:

o Ensure that BookService class has a setter method for BookRepository.

3. Test the Configuration:

o Run the LibraryManagementApplication main class to verify the dependency injection.

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

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

</beans>
```

BookService:

```
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 a new book...");
        bookRepository.save();
    }
}
```

BookRepository:

```
package com.library.repository;
```

```
public class BookRepository {
    public void save() {
        System.out.println("BookRepository: Saving book to
database...");
    }
}
```

MainApp:

```
package com.library;

import com.library.service.BookService;
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");

        BookService bookService = context.getBean("bookService",
BookService.class);
        bookService.addBook();
    }
}
```

Output:

```
OUTPUT   DEBUG CONSOLE   PROBLEMS 33   TEST RESULTS   PORTS   SPELL CHECKER 2   TERMINAL   GITLENS   powershell - library-man

[INFO]
[INFO] -------------------< com.library:library-management >-------------------
[INFO] Building library-management 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- exec:3.1.0:java (default-cli) @ library-management ---
BookService: Adding a new book...
BookRepository: Saving book to database...
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.506 s
[INFO] Finished at: 2025-07-06T15:33:49+05:30
[INFO] ------------------------------------------------------------------------
PS D:\CTS\DN-4.0--Mugilan-Superset-ID-6410310\Week3\library-management>
```

Exercise 4: Creating and Configuring a Maven Project

Scenario:

You need to set up a new Maven project for the library management application and add Spring dependencies.

Steps:

1. Create a New Maven Project:

o Create a new Maven project named LibraryManagement.

2. Add Spring Dependencies in pom.xml:

o Include dependencies for Spring Context, Spring AOP, and Spring WebMVC.

3. Configure Maven Plugins:

o Configure the Maven Compiler Plugin for Java version 1.8 in the pom.xml file.

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-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>1.8</maven.compiler.source>
        <maven.compiler.target>1.8</maven.compiler.target>
    </properties>

    <dependencies>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.36</version>
        </dependency>
```

```
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-aop</artifactId>
            <version>5.3.36</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>5.3.36</version>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.10.1</version>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.codehaus.mojo</groupId>
                <artifactId>exec-maven-plugin</artifactId>
                <version>3.1.0</version>
            </plugin>
        </plugins>
    </build>

</project>
```

Output:

```
OUTPUT   DEBUG CONSOLE   PROBLEMS 45   TEST RESULTS   PORTS   SPELL CHECKER 8   TERMINAL   GITLENS        powershell - library-managements

[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ LibraryManagement ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory D:\CTS\DN-4.0--Mugilan-Superset-ID-6410310\Week3\library-managements\src\main\resources
[INFO]
[INFO] --- compiler:3.10.1:compile (default-compile) @ LibraryManagement ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to D:\CTS\DN-4.0--Mugilan-Superset-ID-6410310\Week3\library-managements\target\classes
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  2.673 s
[INFO] Finished at: 2025-07-06T15:52:24+05:30
[INFO] ------------------------------------------------------------------------
PS D:\CTS\DN-4.0--Mugilan-Superset-ID-6410310\Week3\library-managements>
```

Spring Data JPA - Quick Example:

Properties:

```
spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn
spring.datasource.username=root
spring.datasource.password=Mugilan12!
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

Ormlearnapplication.java:

```
package com.cognizant.ormlearn;
import com.cognizant.ormlearn.model.Country;
import com.cognizant.ormlearn.service.CountryService;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;
import java.util.List;
@SpringBootApplication
public class OrmLearnApplication {
	private static final Logger LOGGER =
LoggerFactory.getLogger(OrmLearnApplication.class);
	public static void main(String[] args) {
		ApplicationContext context = SpringApplication.run(OrmLearnApplication.class,
args);
		LOGGER.info("Inside main");
		CountryService countryService = context.getBean(CountryService.class);
		LOGGER.info("Start");
		testGetAllCountries(countryService);
		LOGGER.info("End");
	}
	public static void testGetAllCountries(CountryService countryService) {
		List<Country> countries = countryService.getAllCountries();
		for (Country country : countries) {
			System.out.println("Country: Code = " + country.getCode() + ", Name = " +
country.getName());
		}
	}
}
```

Country.java:

```
package com.cognizant.ormlearn.model;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.Table;
@Entity
```

```java
@Table(name = "country")
public class Country {
  @Id
  @Column(name = "code")
  private String code;
  @Column(name = "name")
  private String name;
  public String getCode() {
    return code;
  }
  public void setCode(String code) {
    this.code = code;
  }
  public String getName() {
    return name;
  }
  public void setName(String name) {
    this.name = name;
  }
  @Override
  public String toString() {
    return "Country [code=" + code + ", name=" + name + "]";
  }
}
```

CountryService.java:

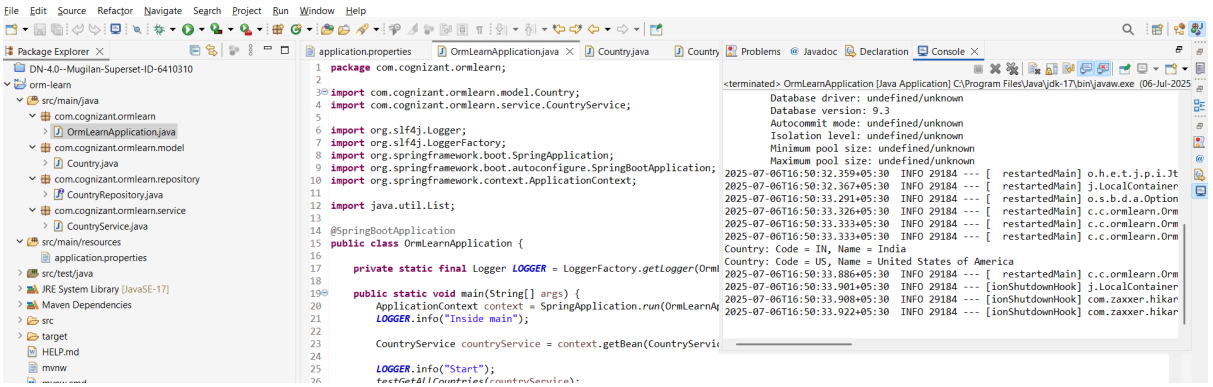
```java
package com.cognizant.ormlearn.service;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import com.cognizant.ormlearn.model.Country;
import com.cognizant.ormlearn.repository.CountryRepository;
@Service
public class CountryService {
  @Autowired
  private CountryRepository countryRepository;
  @Transactional
  public List<Country> getAllCountries() {
    return countryRepository.findAll();
  }
}
```

Countryrepository.java:

```java
package com.cognizant.ormlearn.repository;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import com.cognizant.ormlearn.model.Country;
```

```
@Repository
public interface CountryRepository extends JpaRepository<Country, String> {
}
```

Output:



Difference between JPA, Hibernate and Spring Data JPA:

employee.java:

```
package com.cognizant.ormlearn.model;
import jakarta.persistence.*;
@Entity
@Table(name = "employee")
public class Employee {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private int id;
  private String name;
  private double salary;
  // Getters and Setters
  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 double getSalary() {
     return salary;
  }
  public void setSalary(double salary) {
     this.salary = salary;
```

```
  }
}
```

Employee repository:

```
package com.cognizant.ormlearn.repository;
import org.springframework.data.jpa.repository.JpaRepository;
import com.cognizant.ormlearn.model.Employee;
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {
}
```

EmployeeService.java:

```
package com.cognizant.ormlearn.service;
import com.cognizant.ormlearn.model.Employee;
import com.cognizant.ormlearn.repository.EmployeeRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
@Service
public class EmployeeService {
  @Autowired
  private EmployeeRepository employeeRepository;
  @Transactional
  public void addEmployee(Employee employee) {
     employeeRepository.save(employee);
  }
}
```

Output:

eclipse-workspace - orm-learn/src/main/java/com/cognizant/ormlearn/repository/EmployeeRepository.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Problems Javadoc Declaration Console

```
<terminated> OrmLearnApplication [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe  (06-Jul-2025, 5:12:08 pm – 5:12:19 pm elapsed: 0:00:10.420) [pid: 6960]

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v3.5.3)

25-07-06T17:12:11.182+05:30  INFO 6960 --- [  restartedMain] c.c.ormlearn.OrmLearnApplication         : Starting OrmLearnApplication using Java 17.0.11 with PID 6960 (C:\Users\think\Downloads\orm-learn
25-07-06T17:12:11.188+05:30  INFO 6960 --- [  restartedMain] c.c.ormlearn.OrmLearnApplication         : No active profile set, falling back to 1 default profile: "default"
25-07-06T17:12:11.290+05:30  INFO 6960 --- [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'spring.devtools.add-properties' to 'false' to disable
25-07-06T17:12:12.298+05:30  INFO 6960 --- [  restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
25-07-06T17:12:12.421+05:30  INFO 6960 --- [  restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 107 ms. Found 2 JPA repository interfaces.
25-07-06T17:12:13.125+05:30  INFO 6960 --- [  restartedMain] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
25-07-06T17:12:13.985+05:30  INFO 6960 --- [  restartedMain] com.zaxxer.hikari.pool.HikariPool        : HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionImpl@75044d35
25-07-06T17:12:13.989+05:30  INFO 6960 --- [  restartedMain] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
25-07-06T17:12:14.070+05:30  INFO 6960 --- [  restartedMain] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
25-07-06T17:12:14.232+05:30  INFO 6960 --- [  restartedMain] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 6.6.18.Final
25-07-06T17:12:14.347+05:30  INFO 6960 --- [  restartedMain] o.h.c.internal.RegionFactoryInitiator    : HHH000026: Second-level cache disabled
25-07-06T17:12:15.050+05:30  INFO 6960 --- [  restartedMain] o.s.o.j.p.SpringPersistenceUnitInfo      : No LoadTimeWeaver setup: ignoring JPA class transformer
25-07-06T17:12:15.338+05:30  INFO 6960 --- [  restartedMain] org.hibernate.orm.connections.pooling    : HHH10001005: Database info:
      Database JDBC URL [Connecting through datasource 'HikariDataSource (HikariPool-1)']
      Database driver: undefined/unknown
      Database version: 9.3
      Autocommit mode: undefined/unknown
      Isolation level: undefined/unknown
      Minimum pool size: undefined/unknown
      Maximum pool size: undefined/unknown
25-07-06T17:12:16.923+05:30  INFO 6960 --- [  restartedMain] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to enable JTA plat
25-07-06T17:12:16.931+05:30  INFO 6960 --- [  restartedMain] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
25-07-06T17:12:17.894+05:30  INFO 6960 --- [  restartedMain] o.s.b.d.a.OptionalLiveReloadServer       : LiveReload server is running on port 35729
25-07-06T17:12:17.935+05:30  INFO 6960 --- [  restartedMain] c.c.ormlearn.OrmLearnApplication         : Started OrmLearnApplication in 7.591 seconds (process running for 8.486)
25-07-06T17:12:17.942+05:30  INFO 6960 --- [  restartedMain] c.c.ormlearn.OrmLearnApplication         : Inside main
25-07-06T17:12:17.942+05:30  INFO 6960 --- [  restartedMain] c.c.ormlearn.OrmLearnApplication         : Start
untry: Code = IN, Name = India
untry: Code = US, Name = United States of America
25-07-06T17:12:18.470+05:30  INFO 6960 --- [  restartedMain] c.c.ormlearn.OrmLearnApplication         : End
25-07-06T17:12:18.479+05:30  INFO 6960 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persistence unit 'default'
25-07-06T17:12:18.486+05:30  INFO 6960 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown initiated...
25-07-06T17:12:18.503+05:30  INFO 6960 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown completed.
```