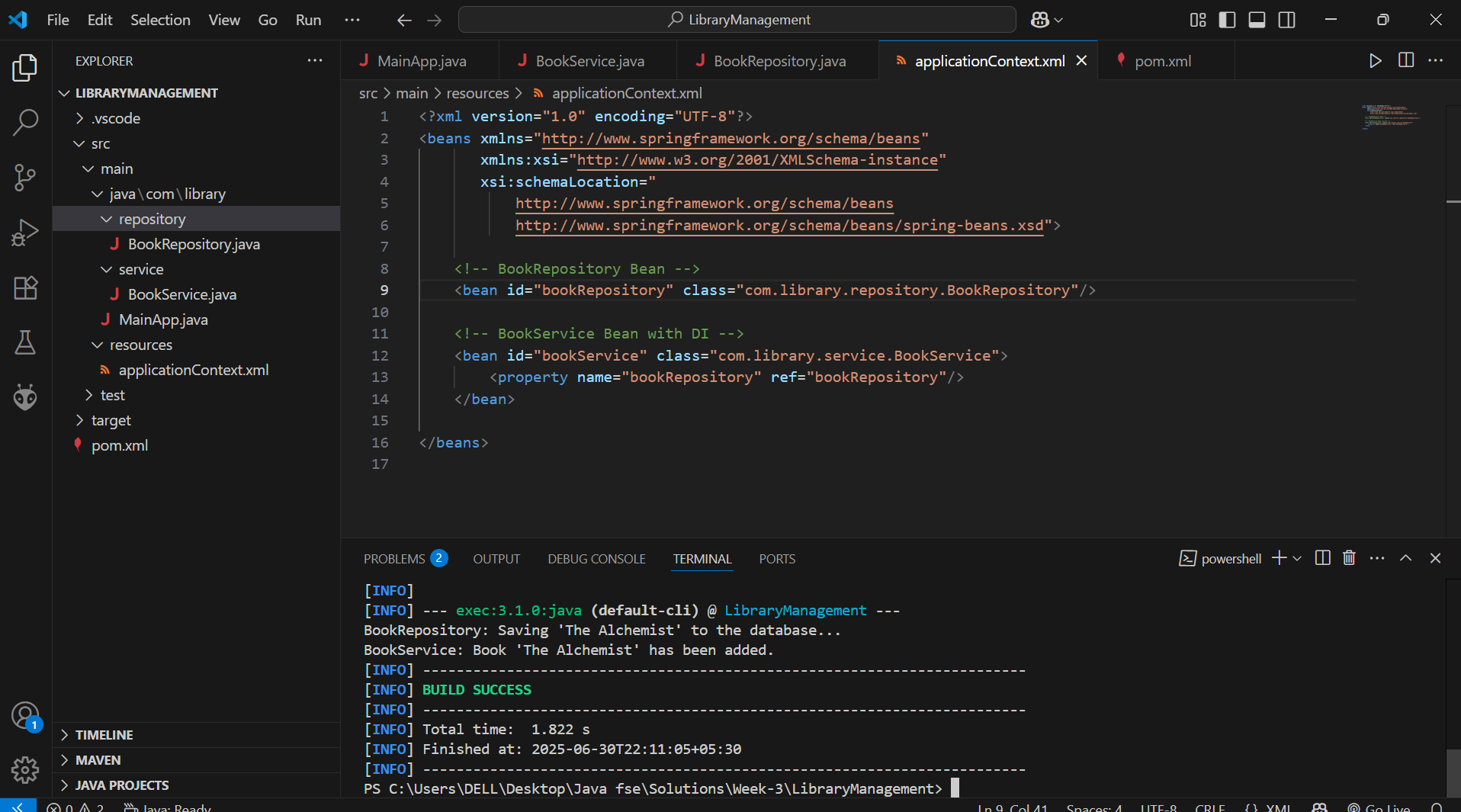
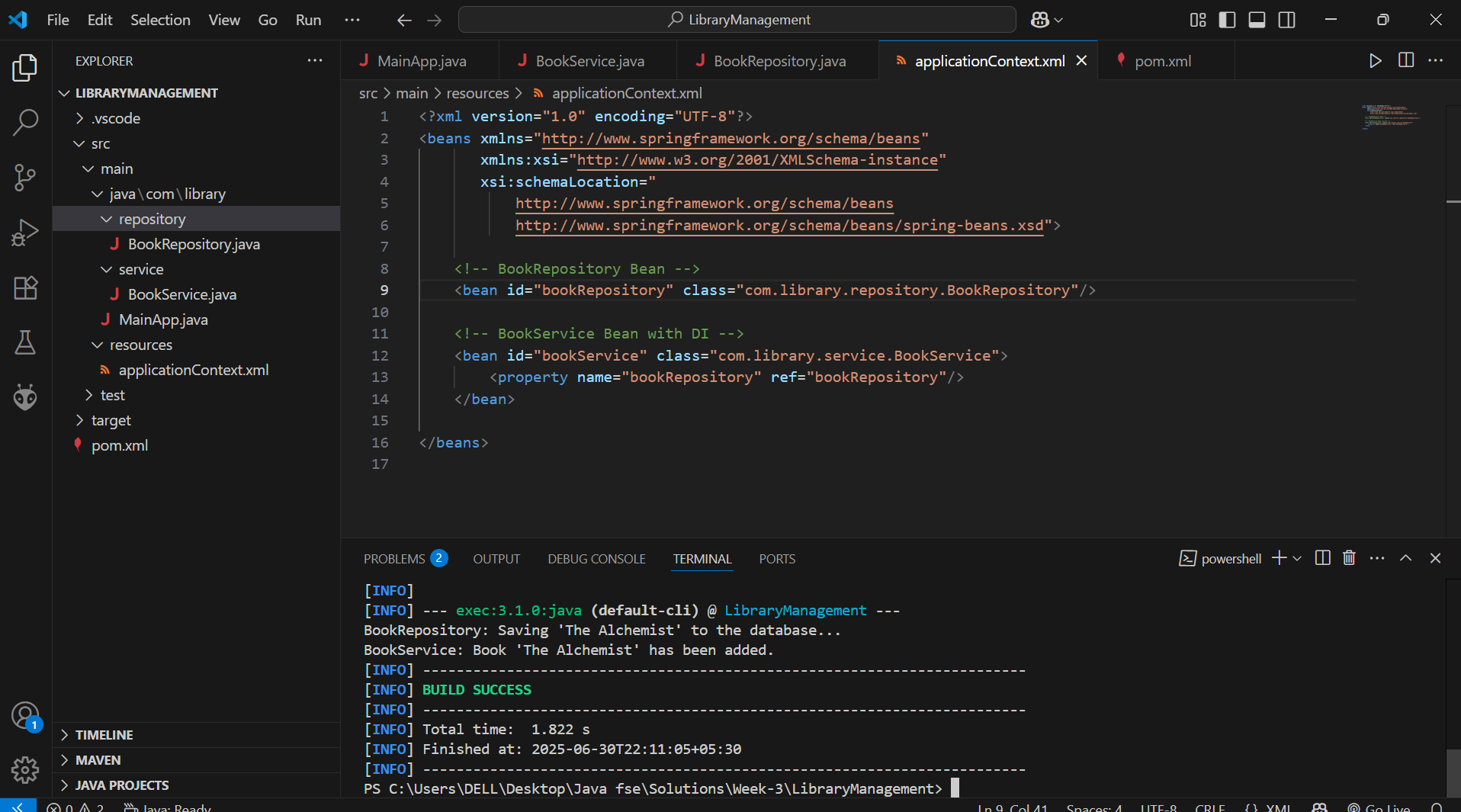
Exercise 1: Configuring a Basic Spring Application:



Exercise 2: Implementing Dependency Injection:



Exercise 4: Creating and Configuring a Maven Project:

Dependency adding in pom.xml file:

<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>

<dependencies>

        <dependency>

    <groupId>org.springframework</groupId>

    <artifactId>spring-context</artifactId>

    <version>6.2.8</version>

</dependency>

<dependency>

    <groupId>org.springframework</groupId>

    <artifactId>spring-aop</artifactId>

    <version>6.2.8</version>

</dependency>

<dependency>

    <groupId>org.springframework</groupId>

    <artifactId>spring-webmvc</artifactId>

    <version>6.2.8</version>

</dependency>

    </dependencies>

    <build>

        <plugins>

            <plugin>

                <groupId>org.codehaus.mojo</groupId>

                <artifactId>exec-maven-plugin</artifactId>

                <version>3.1.0</version>

                <configuration>

                    <mainClass>com.library.MainApp</mainClass>

                </configuration>

            </plugin>

             <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>

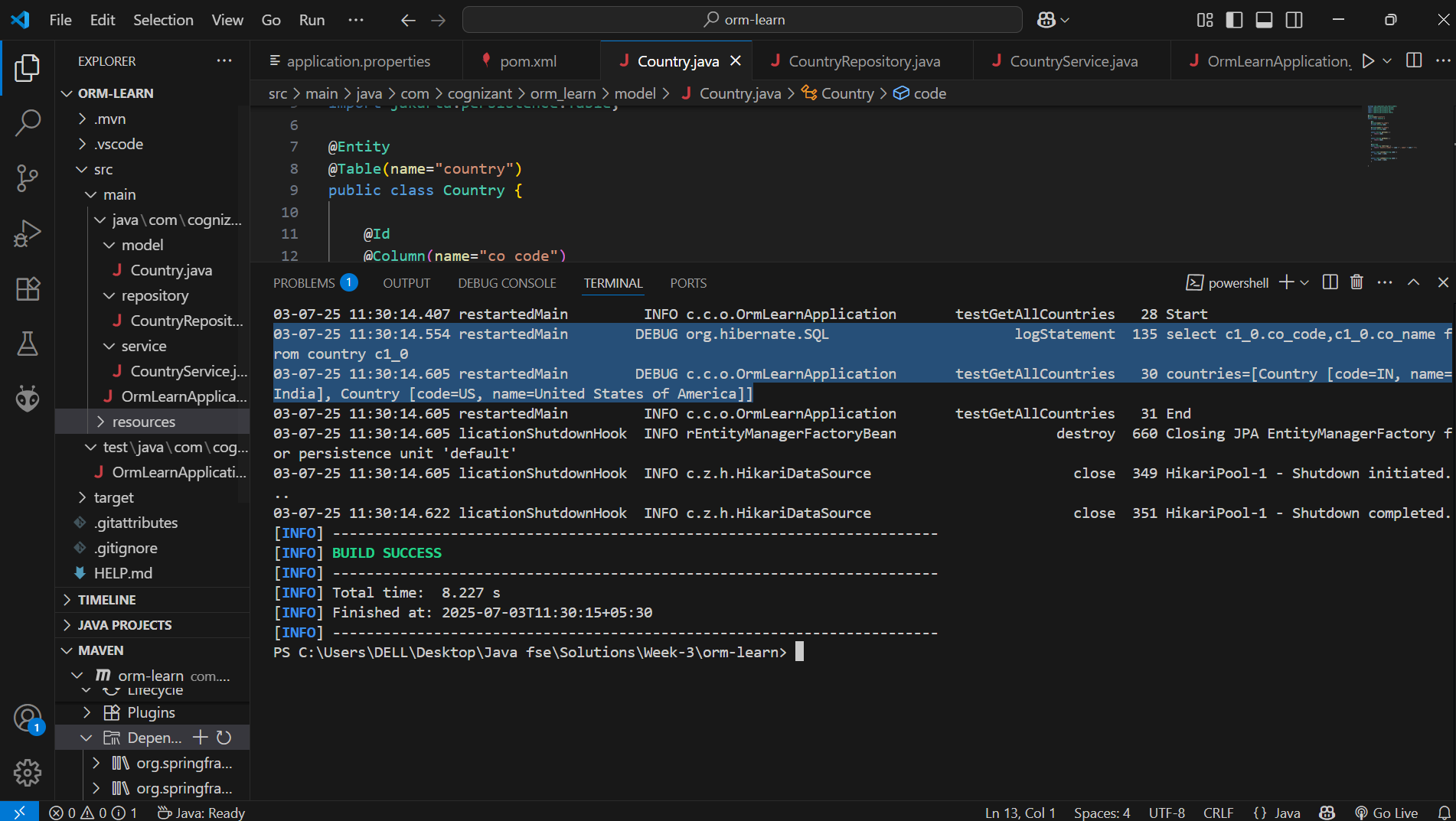
        </plugin>

        </plugins>

</build>

</project>

Spring Data JPA - Quick Example:



Difference between JPA, Hibernate and Spring Data JPA:

JPA/Hibernate/Spring Data JPA

A standard or the specification for ORM (Object Relational Mapping). It provides a common interfaces for interacting with relational databases using Java objects.

It just a set of interfaces, not an implementation.

Whereas Hibernate is considered as the most popular implementation of JPA.

Hibernate has feature that are far beyond than caching, custom SQL support, interceptors and event listeners.

It is the engine that actually interacts with your database.

On top of JPA and Hibernate, Spring Data JPA is used. It is a spring based abstraction over JPA that dramatically reduces boilerplate code/repository code.

JPA:

@Entity

@Table(name="country")

public class Country {

    @Id

    @Column(name="co\_code")

    private String code;

    @Column(name="co\_name")

    private String name;

    public String getCode() {

        return code;

    }

Hibernate:

import org.hibernate.\*;

import org.hibernate.cfg.Configuration;

import java.util.List;

public class HibernateExample {

    public static void main(String[] args) {

        SessionFactory factory = new Configuration().configure().buildSessionFactory();

        Session session = factory.openSession();

        session.beginTransaction();

        List<Country> countries = session.createQuery("FROM Country", Country.class).list();

        session.getTransaction().commit();

        countries.forEach(System.out::println);

        session.close();

        factory.close();

    }

}

Spring Data JPA:

package com.cognizant.orm\_learn.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.cognizant.orm\_learn.model.Country;

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

}