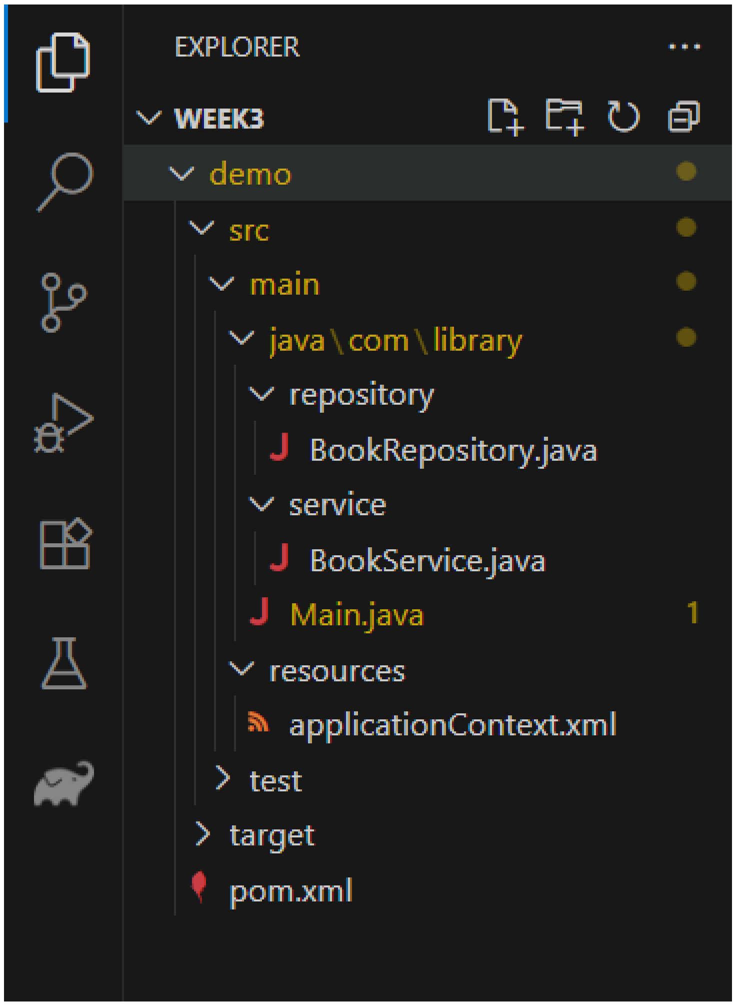
WEEK-3 HANDS ON

Spring Core_Maven

Exercise 1: Configuring a Basic Spring Application

- Created a Maven Project in VS Code using the steps.
    - Ctrl + Shift + P
    - Create java Project
    - Maven
    - No Archetype
    - com.library

Folder Structure



```
EXPLORER
WEEK3
  demo
    src
      main
        java\com\library
          repository
            BookRepository.java
          service
            BookService.java
          Main.java
        resources
          applicationContext.xml
      test
    target
    pom.xml
```

pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<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>demo</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>17</maven.compiler.source>
        <maven.compiler.target>17</maven.compiler.target>
    </properties>

    <dependencies>
        <!-- Spring Core dependency -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>
    </dependencies>

</project>
```

applicationContext.xml

```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">

    <!-- Define BookRepository bean -->
    <bean id="bookRepository"
class="com.library.repository.BookRepository" />
```

```xml
    <!-- Define BookService bean and inject bookRepository -->
    <bean id="bookService" class="com.library.service.BookService">
        <property name="bookRepository" ref="bookRepository" />
    </bean>

</beans>
```

BookService.java

```java
package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private BookRepository bookRepository;

    public void setBookRepository(BookRepository bookRepository) {
        // System.out.println("Setter injection Called");
        this.bookRepository = bookRepository;
    }

    public void addBook(String title) {
        System.out.println("Adding book...");
        bookRepository.saveBook(title);
    }
}
```

BookRepository.java

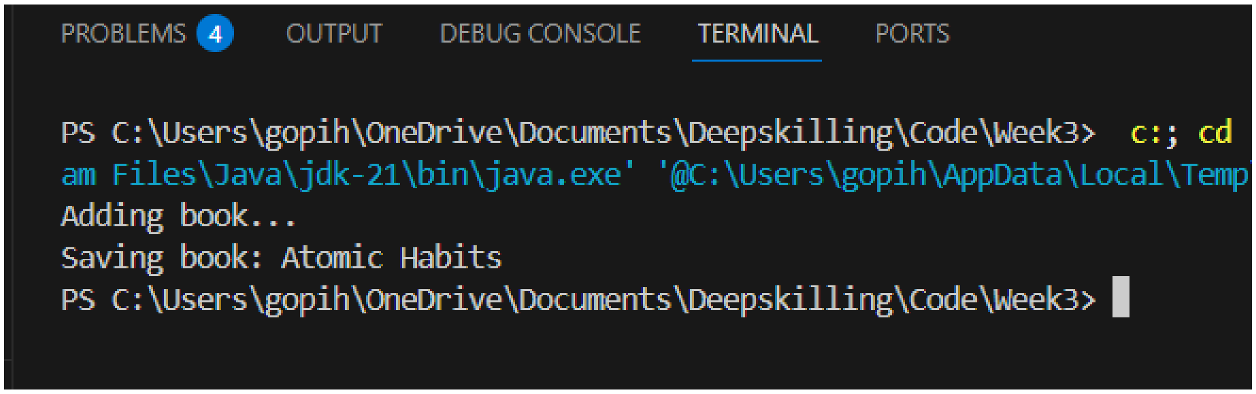
```java
package com.library.repository;

public class BookRepository {
    public void saveBook(String title) {
        System.out.println("Saving book: " + title);
    }
}
```

Main.java

```java
package com.library;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class Main {
    public static void main(String[] args) {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");

        BookService service = (BookService)
context.getBean("bookService");
        service.addBook("Atomic Habits");
    }
}
```

Output:

The Test needs to verify whether the application is being created and the BookService and BookRepository are being executed or not. The logs from both the classes show they are working.



```
PS C:\Users\gopih\OneDrive\Documents\Deepskilling\Code\Week3>  c:; cd
am Files\Java\jdk-21\bin\java.exe' '@C:\Users\gopih\AppData\Local\Temp
Adding book...
Saving book: Atomic Habits
PS C:\Users\gopih\OneDrive\Documents\Deepskilling\Code\Week3>
```

Exercise 2: Implementing Dependency Injection

- Added a log to check whether the dependencies are being injected or not in ‘BookService.java’
- The other classes remain the same.

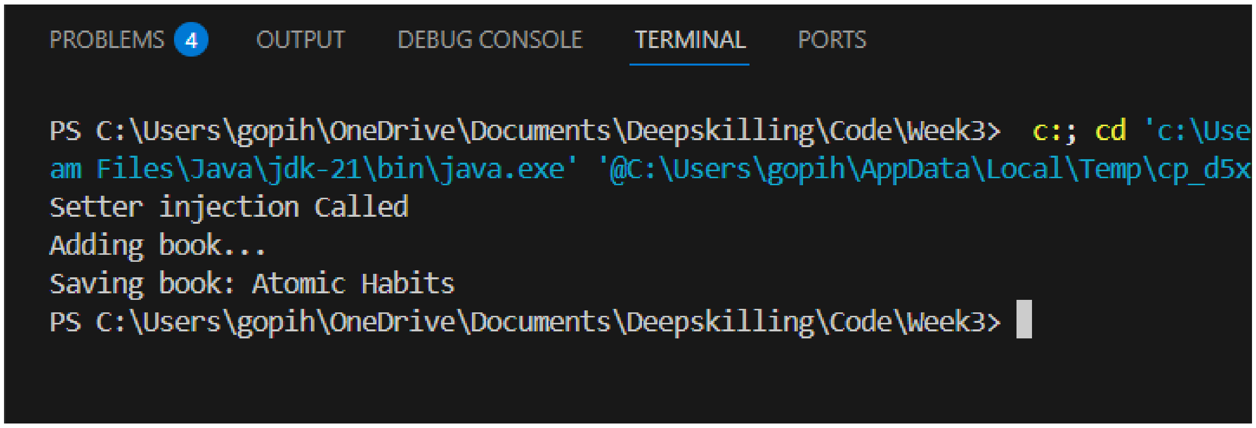
BookService.java

```java
package com.library.service;

import com.library.repository.BookRepository;
```

```java
public class BookService {
    private BookRepository bookRepository;

    // Setter for injection
    public void setBookRepository(BookRepository bookRepository) {
        System.out.println("Setter injection Called");
        this.bookRepository = bookRepository;
    }

    public void addBook(String title) {
        System.out.println("Adding book...");
        bookRepository.saveBook(title);
    }
}
```

Output:

During the injection of the dependency “BookRepository” to the “BookService”, the setBookRepository is being called and it is being injected to the BookService. The log proves the working of this.



```
PS C:\Users\gopih\OneDrive\Documents\Deepskilling\Code\Week3>  c:; cd 'c:\Use
am Files\Java\jdk-21\bin\java.exe' '@C:\Users\gopih\AppData\Local\Temp\cp_d5x
Setter injection Called
Adding book...
Saving book: Atomic Habits
PS C:\Users\gopih\OneDrive\Documents\Deepskilling\Code\Week3>
```

Exercise 4: Creating and Configuring a Maven Project

- Created a Maven Project in VS Code using the following steps:
    - Ctrl + Shift + P
    - Create java Project
    - Maven
    - No Archetype
    - com.library
    - Project Name: Library Management

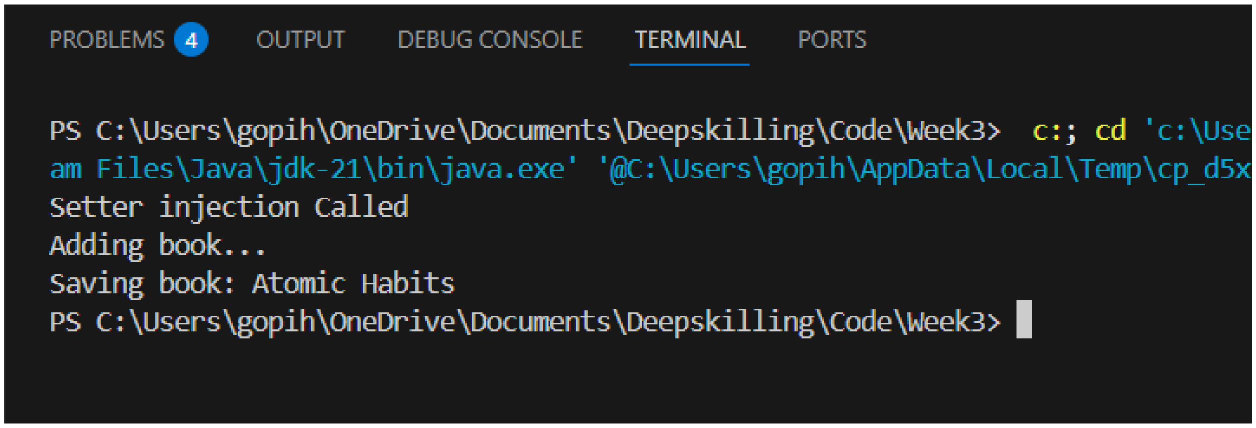
- Modified the “pom.xml” file to include dependencies of Spring Context, Spring AOP and Spring WebMVC as well as the Maven compiler plugin.

pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<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>
        <!-- Spring Context (core container + DI) -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.3.34</version>
        </dependency>

        <!-- Spring AOP (for aspect-oriented programming) -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-aop</artifactId>
            <version>5.3.34</version>
        </dependency>

        <!-- Spring WebMVC (for servlet-based web apps) -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>5.3.34</version>
        </dependency>

        <!-- Required for AOP -->
        <dependency>
            <groupId>org.aspectj</groupId>
            <artifactId>aspectjweaver</artifactId>
            <version>1.9.21</version>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <!-- Configure Maven Compiler Plugin -->
            <plugin>
                <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:

The project gets compiled, then executed and got the following output without any exceptions or errors.

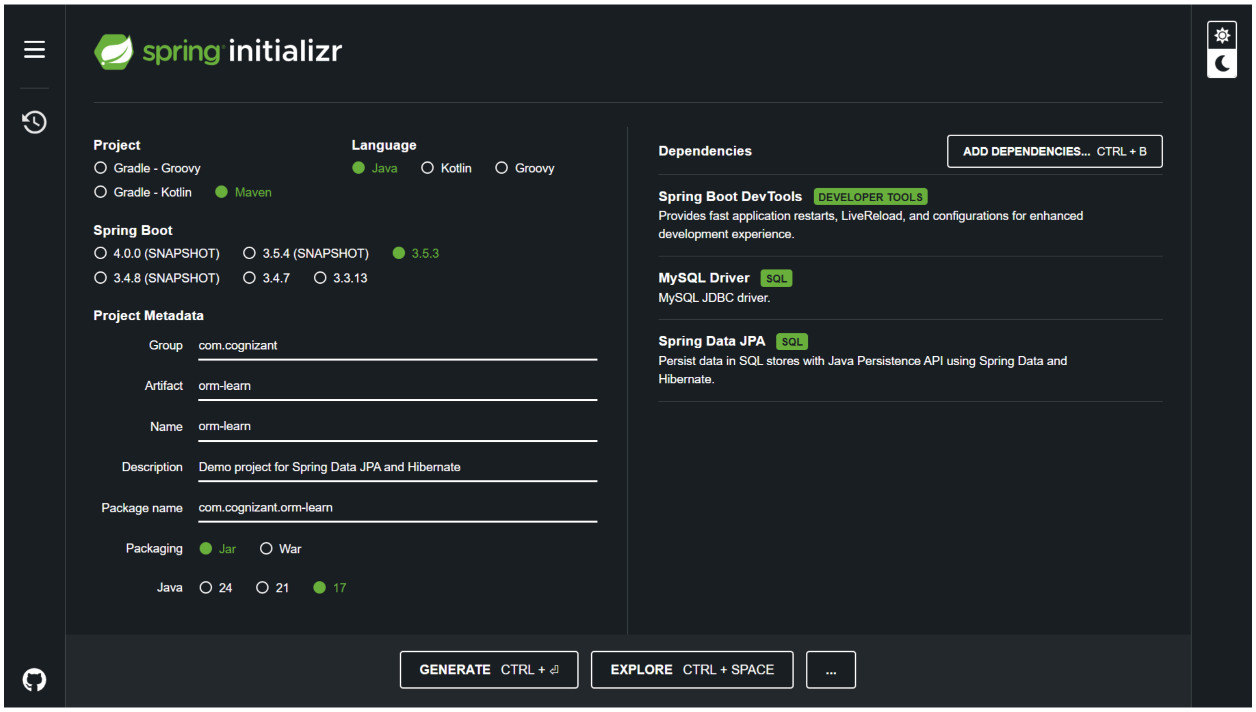


```
PS C:\Users\gopih\OneDrive\Documents\Deepskilling\Code\Week3>  c:; cd 'c:\Use
am Files\Java\jdk-21\bin\java.exe' '@C:\Users\gopih\AppData\Local\Temp\cp_d5x
Setter injection Called
Adding book...
Saving book: Atomic Habits
PS C:\Users\gopih\OneDrive\Documents\Deepskilling\Code\Week3>
```

Spring Data JPA - Quick Example

- Installed the pre-requisites: MySQL, Eclipse and Maven
- Utilized Spring Initializer to configure the Spring application with the given following requirements
    - Project: Maven
    - Language: Java 17
    - Version: 3.5.3
    - Custom names
    - Dependencies: Spring Boot Dev Tools, MySQL Driver, Spring Data JPA
- Created a database schema named ‘ormlearn’ and created a table named ‘country’ and added a few sample data using MySQL Command Line Client.
- Modified the application.properties file in src/main/resources folder. Included the configuration details regarding:
    - Spring Application
    - Framework, logging and Log pattern
    - Database Configuration
    - Hibernate Configuration
- Added the logger to the main class and tested for the logger
- Modified the src/main folder in such a way that it efficiently implements three-tier architecture
    - **Model:** Used to represent real-world entities that are stored in database. Created Country.java in src/main/model folder which denotes the class model. Annotations were used that are needed for Object Relational Mapping (ORM)
    - **Repository:** Used to handle direct interaction with the database. Contains no logic, only determines what kind of data access is possible. Created a “CountryRepository.java” class that extends JpaRepository class
    - **Service:** Contains the actual business logic. Created a “CountryService.java” class to provide the services related to Country class.
- Application has been tested in OrmApplication.java

Spring Initializr



Project

Gradle - Groovy / Gradle - Kotlin / [✓ Maven]

Language

[✓ Java] / Kotlin / Groovy

Spring Boot

4.0.0 (SNAPSHOT) / 3.5.4 (SNAPSHOT) / [✓ 3.5.3] / 3.4.8 (SNAPSHOT) / 3.4.7 / 3.3.13

Project Metadata

Group	com.cognizant
Artifact	orm-learn
Name	orm-learn
Description	Demo project for Spring Data JPA and Hibernate
Package name	com.cognizant.orm-learn
Packaging	[✓ Jar] / War
Java	24 / 21 / [✓ 17]

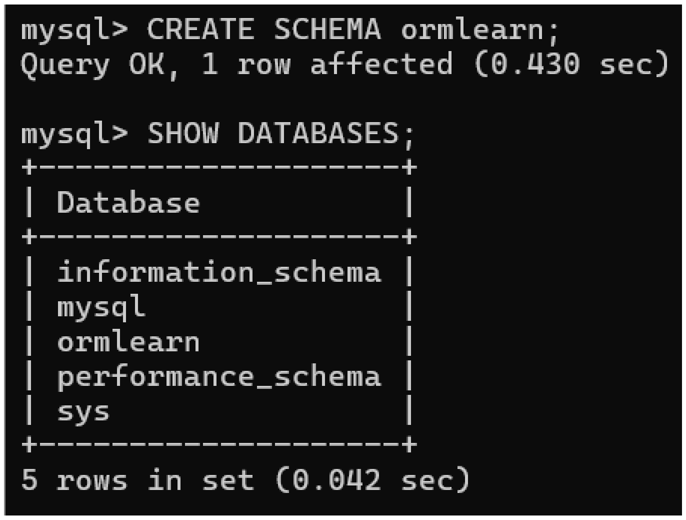
Dependencies

Spring Boot DevTools DEVELOPER TOOLS
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

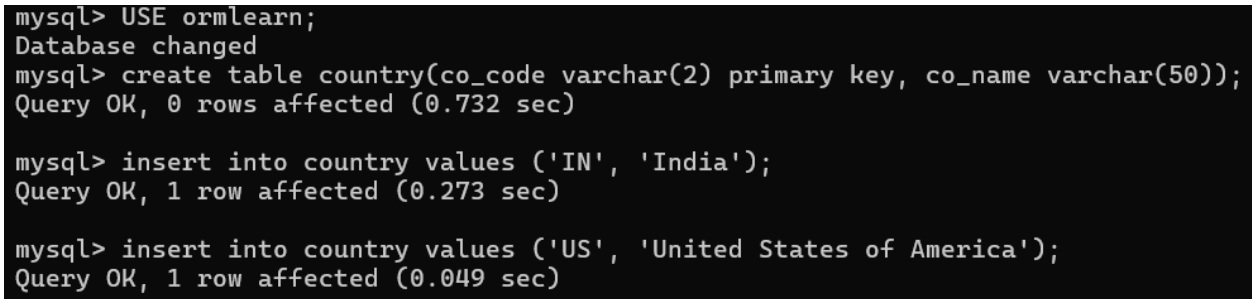
MySQL Driver SQL
MySQL JDBC driver.

Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

MySQL



```
mysql> CREATE SCHEMA ormlearn;
Query OK, 1 row affected (0.430 sec)

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| ormlearn           |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.042 sec)
```



```
mysql> USE ormlearn;
Database changed
mysql> create table country(co_code varchar(2) primary key, co_name varchar(50));
Query OK, 0 rows affected (0.732 sec)

mysql> insert into country values ('IN', 'India');
Query OK, 1 row affected (0.273 sec)

mysql> insert into country values ('US', 'United States of America');
Query OK, 1 row affected (0.049 sec)
```

Country.java

```java
package com.cognizant.orm_learn.model;
import jakarta.persistence.Entity;
import jakarta.persistence.Table;
import jakarta.persistence.Column;
import jakarta.persistence.Id;
@Entity
@Table(name="country")
public class Country{
        @Id
        @Column(name="co_code")
        private String code;

        @Column(name="co_name")
        private String name;

        public void setCode(String code) {
                this.code = code;
        }

        public void setName(String code) {
                this.name = name;
        }

        public String getCode() {
                return this.code;
        }

        public String getName() {
                return this.name;
        }
}
```

CountryRepository.java

```java
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> {

}
```

CountryService.java

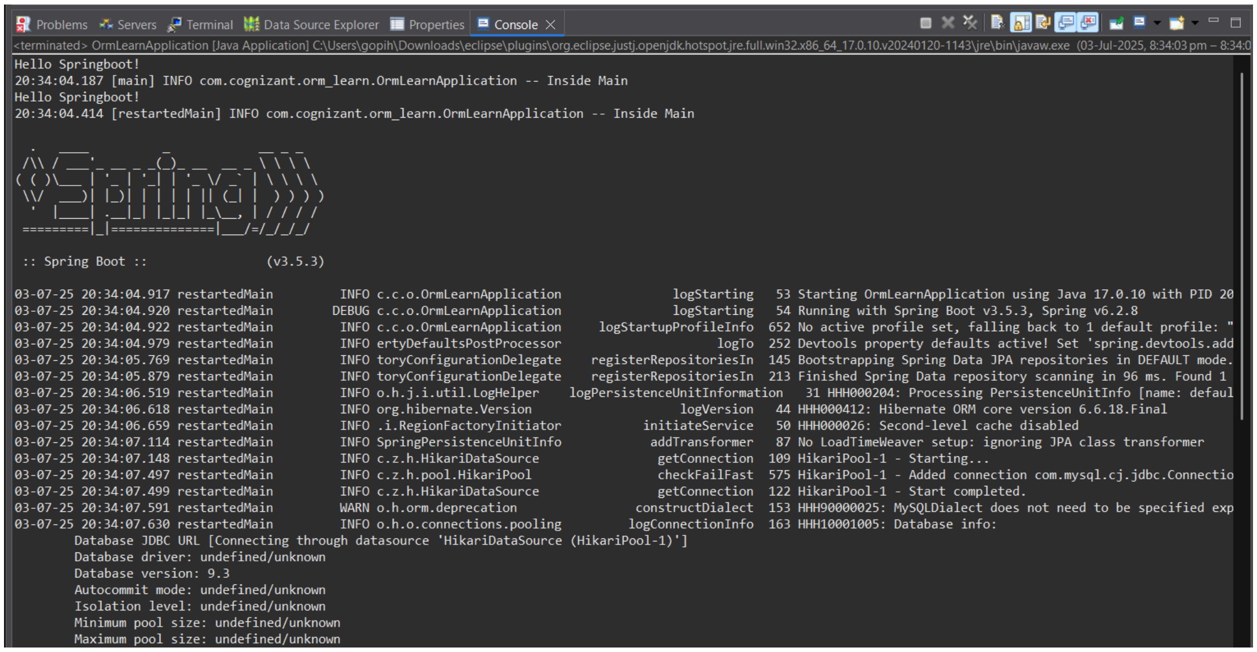
```java
package com.cognizant.orm_learn.service;
import com.cognizant.orm_learn.repository.CountryRepository;
import com.cognizant.orm_learn.model.Country;
import org.springframework.stereotype.Service;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.transaction.annotation.Transactional;
import java.util.List;
@Service
public class CountryService{

        @Autowired
        private CountryRepository countryRepository;

        @Transactional
        public List<Country> getAllCountries(){
                return countryRepository.findAll();
        }
}
```

OrmApplication.java

```java
package com.cognizant.orm_learn;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import java.util.List;
import com.cognizant.orm_learn.model.Country;
import com.cognizant.orm_learn.service.CountryService;
```

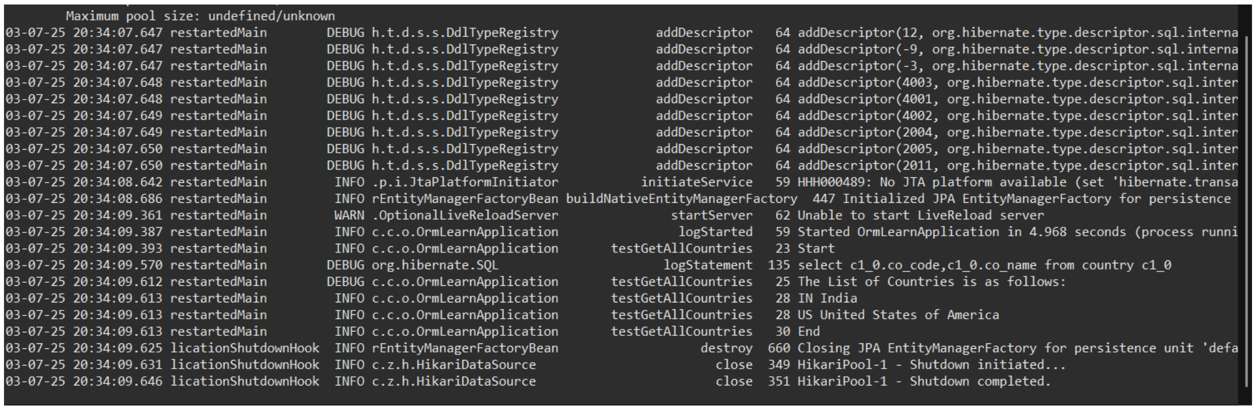
```
@SpringBootApplication
public class OrmLearnApplication {

    private static final Logger logger = LoggerFactory.getLogger(OrmLearnApplication.class);

    private static CountryService countryService;

    private static void testGetAllCountries() {
        logger.info("Start");
        List<Country> countries = countryService.getAllCountries();
        logger.debug("The List of Countries is as follows: ");
        for(Country country: countries) {
            String str = country.getCode()+" "+country.getName();
            logger.info(str);
        }
        logger.info("End");
    }
    public static void main(String[] args) {
//        SpringApplication.run(OrmLearnApplication.class, args);
        System.out.println("Hello Springboot!");
        logger.info("Inside Main");

        ApplicationContext context = SpringApplication.run(OrmLearnApplication.class,
args);
        countryService = context.getBean(CountryService.class);
        testGetAllCountries();

    }
}
```

Output:

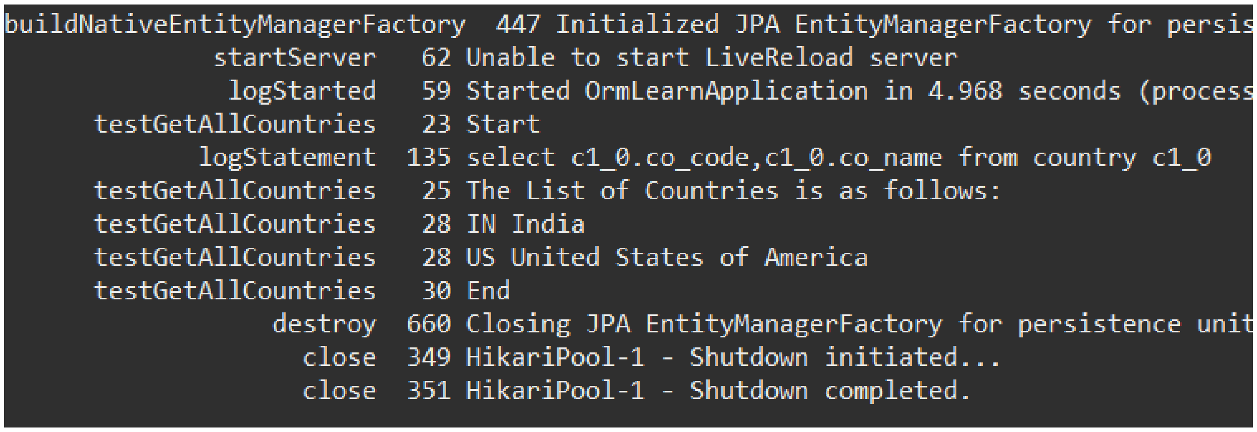


Problems Servers Terminal Data Source Explorer Properties Console

```
<terminated> OrmLearnApplication [Java Application] C:\Users\gopih\Downloads\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.10.v20240120-1143\jre\bin\javaw.exe (03-Jul-2025, 8:34:03 pm – 8:34:0
Hello Springboot!
20:34:04.187 [main] INFO com.cognizant.orm_learn.OrmLearnApplication -- Inside Main
Hello Springboot!
20:34:04.414 [restartedMain] INFO com.cognizant.orm_learn.OrmLearnApplication -- Inside Main

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

 :: Spring Boot ::                (v3.5.3)

03-07-25 20:34:04.917 restartedMain        INFO c.c.o.OrmLearnApplication               logStarting   53 Starting OrmLearnApplication using Java 17.0.10 with PID 20
03-07-25 20:34:04.920 restartedMain       DEBUG c.c.o.OrmLearnApplication               logStarting   54 Running with Spring Boot v3.5.3, Spring v6.2.8
03-07-25 20:34:04.922 restartedMain        INFO c.c.o.OrmLearnApplication     logStartupProfileInfo  652 No active profile set, falling back to 1 default profile: "
03-07-25 20:34:04.979 restartedMain        INFO ertyDefaultsPostProcessor                     logTo  252 Devtools property defaults active! Set 'spring.devtools.add
03-07-25 20:34:05.769 restartedMain        INFO toryConfigurationDelegate    registerRepositoriesIn  145 Bootstrapping Spring Data JPA repositories in DEFAULT mode.
03-07-25 20:34:05.879 restartedMain        INFO toryConfigurationDelegate    registerRepositoriesIn  213 Finished Spring Data repository scanning in 96 ms. Found 1
03-07-25 20:34:06.519 restartedMain        INFO o.h.j.i.util.LogHelper   logPersistenceUnitInformation   31 HHH000204: Processing PersistenceUnitInfo [name: defaul
03-07-25 20:34:06.618 restartedMain        INFO org.hibernate.Version                    logVersion   44 HHH000412: Hibernate ORM core version 6.6.18.Final
03-07-25 20:34:06.659 restartedMain        INFO .i.RegionFactoryInitiator           initiateService   50 HHH000026: Second-level cache disabled
03-07-25 20:34:07.114 restartedMain        INFO SpringPersistenceUnitInfo            addTransformer   87 No LoadTimeWeaver setup: ignoring JPA class transformer
03-07-25 20:34:07.148 restartedMain        INFO c.z.h.HikariDataSource                getConnection  109 HikariPool-1 - Starting...
03-07-25 20:34:07.497 restartedMain        INFO c.z.h.pool.HikariPool                 checkFailFast  575 HikariPool-1 - Added connection com.mysql.cj.jdbc.Connectio
03-07-25 20:34:07.499 restartedMain        INFO c.z.h.HikariDataSource                getConnection  122 HikariPool-1 - Start completed.
03-07-25 20:34:07.591 restartedMain        WARN o.h.orm.deprecation                constructDialect  153 HHH90000025: MySQLDialect does not need to be specified exp
03-07-25 20:34:07.630 restartedMain        INFO o.h.o.connections.pooling         logConnectionInfo  163 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
```



```
        Maximum pool size: undefined/unknown
03-07-25 20:34:07.647 restartedMain       DEBUG h.t.d.s.s.DdlTypeRegistry             addDescriptor   64 addDescriptor(12, org.hibernate.type.descriptor.sql.interna
03-07-25 20:34:07.647 restartedMain       DEBUG h.t.d.s.s.DdlTypeRegistry             addDescriptor   64 addDescriptor(-9, org.hibernate.type.descriptor.sql.interna
03-07-25 20:34:07.647 restartedMain       DEBUG h.t.d.s.s.DdlTypeRegistry             addDescriptor   64 addDescriptor(-3, org.hibernate.type.descriptor.sql.interna
03-07-25 20:34:07.648 restartedMain       DEBUG h.t.d.s.s.DdlTypeRegistry             addDescriptor   64 addDescriptor(4003, org.hibernate.type.descriptor.sql.inter
03-07-25 20:34:07.648 restartedMain       DEBUG h.t.d.s.s.DdlTypeRegistry             addDescriptor   64 addDescriptor(4001, org.hibernate.type.descriptor.sql.inter
03-07-25 20:34:07.649 restartedMain       DEBUG h.t.d.s.s.DdlTypeRegistry             addDescriptor   64 addDescriptor(4002, org.hibernate.type.descriptor.sql.inter
03-07-25 20:34:07.649 restartedMain       DEBUG h.t.d.s.s.DdlTypeRegistry             addDescriptor   64 addDescriptor(2004, org.hibernate.type.descriptor.sql.inter
03-07-25 20:34:07.650 restartedMain       DEBUG h.t.d.s.s.DdlTypeRegistry             addDescriptor   64 addDescriptor(2005, org.hibernate.type.descriptor.sql.inter
03-07-25 20:34:07.650 restartedMain       DEBUG h.t.d.s.s.DdlTypeRegistry             addDescriptor   64 addDescriptor(2011, org.hibernate.type.descriptor.sql.inter
03-07-25 20:34:08.642 restartedMain        INFO .p.i.JtaPlatformInitiator           initiateService   59 HHH000489: No JTA platform available (set 'hibernate.transa
03-07-25 20:34:08.686 restartedMain        INFO rEntityManagerFactoryBean buildNativeEntityManagerFactory  447 Initialized JPA EntityManagerFactory for persistence
03-07-25 20:34:09.361 restartedMain        WARN .OptionalLiveReloadServer               startServer   62 Unable to start LiveReload server
03-07-25 20:34:09.387 restartedMain        INFO c.c.o.OrmLearnApplication               logStarted   59 Started OrmLearnApplication in 4.968 seconds (process runni
03-07-25 20:34:09.393 restartedMain        INFO c.c.o.OrmLearnApplication       testGetAllCountries   23 Start
03-07-25 20:34:09.570 restartedMain       DEBUG org.hibernate.SQL                      logStatement  135 select c1_0.co_code,c1_0.co_name from country c1_0
03-07-25 20:34:09.612 restartedMain       DEBUG c.c.o.OrmLearnApplication       testGetAllCountries   25 The List of Countries is as follows:
03-07-25 20:34:09.613 restartedMain        INFO c.c.o.OrmLearnApplication       testGetAllCountries   28 IN India
03-07-25 20:34:09.613 restartedMain        INFO c.c.o.OrmLearnApplication       testGetAllCountries   28 US United States of America
03-07-25 20:34:09.613 restartedMain        INFO c.c.o.OrmLearnApplication       testGetAllCountries   30 End
03-07-25 20:34:09.625 licationShutdownHook  INFO rEntityManagerFactoryBean                  destroy  660 Closing JPA EntityManagerFactory for persistence unit 'defa
03-07-25 20:34:09.631 licationShutdownHook  INFO c.z.h.HikariDataSource                       close  349 HikariPool-1 - Shutdown initiated...
03-07-25 20:34:09.646 licationShutdownHook  INFO c.z.h.HikariDataSource                       close  351 HikariPool-1 - Shutdown completed.
```



```
buildNativeEntityManagerFactory  447 Initialized JPA EntityManagerFactory for persis
            startServer   62 Unable to start LiveReload server
             logStarted   59 Started OrmLearnApplication in 4.968 seconds (process
    testGetAllCountries   23 Start
           logStatement  135 select c1_0.co_code,c1_0.co_name from country c1_0
    testGetAllCountries   25 The List of Countries is as follows:
    testGetAllCountries   28 IN India
    testGetAllCountries   28 US United States of America
    testGetAllCountries   30 End
                destroy  660 Closing JPA EntityManagerFactory for persistence unit
                  close  349 HikariPool-1 - Shutdown initiated...
                  close  351 HikariPool-1 - Shutdown completed.
```

Difference between JPA, Hibernate and Spring Data JPA

- Understood the difference between JPA, Hibernate and Spring Data JPA
- **JPA:**
    - JPA stands for Java Persistence API.
    - It provides standards rules/annotations required for Object Relational Mapping.
    - It is a standard defined by Jakarta EE / Java EE.
    - Examples: @Entity, @Id, @Column, @Table, etc.
- **Hibernate:**
    - It is an implementation of JPA
    - Provides actual code to perform ORM as per JPA's rules.
    - Can be implemented using Session Factory, Sessions, Queries, Transactions and Contexts. Queries are database independent
    - It is created by Red Hat.
    - Features: Caching, Lazy loading, HQL, Schema generation, etc.
- **Spring Data JPA**
    - It is a part of Spring Framework
    - Makes JPA easier to use with Spring
    - It is implemented using JpaRepository that provides data that can be accessible.