# **WEEK 3 spring DATA JPA WITH SRING BOOT HIBERNATE HANDS ON**

**SUBMITTED BY :-**

NAME – AYUSH SAHOO

SUPERSET ID – 6372243

COURSE – DIGITAL NURTURE 4.0 JAVA FSE

EMAIL – [ayush.sahoo003@gmail.com](mailto:ayush.sahoo003@gmail.com)

COLLEGE – C.V. RAMAN GLOBAL UNIVERSITY

**-----------------NOTE: I HAVE ALSO DONE ADDITIONAL IMPORTANT HANDS ON -------------------**

**MANDATORY HANDS ON**

**EXERCISE 1:**

**Spring Data JPA - Quick Example   
  
Software Pre-requisites**

* **MySQL Server 8.0**
* **MySQL Workbench 8**
* **Eclipse IDE for Enterprise Java Developers 2019-03 R**
* **Maven 3.6.2**

**Create a Eclipse Project using Spring Initializr**

**Execute the OrmLearnApplication and check in log if main method is called.**

**Country table creation**

**Persistence Class - com.cognizant.orm-learn.model.Country**

**Repository Class - com.cognizant.orm-learn.CountryRepository**

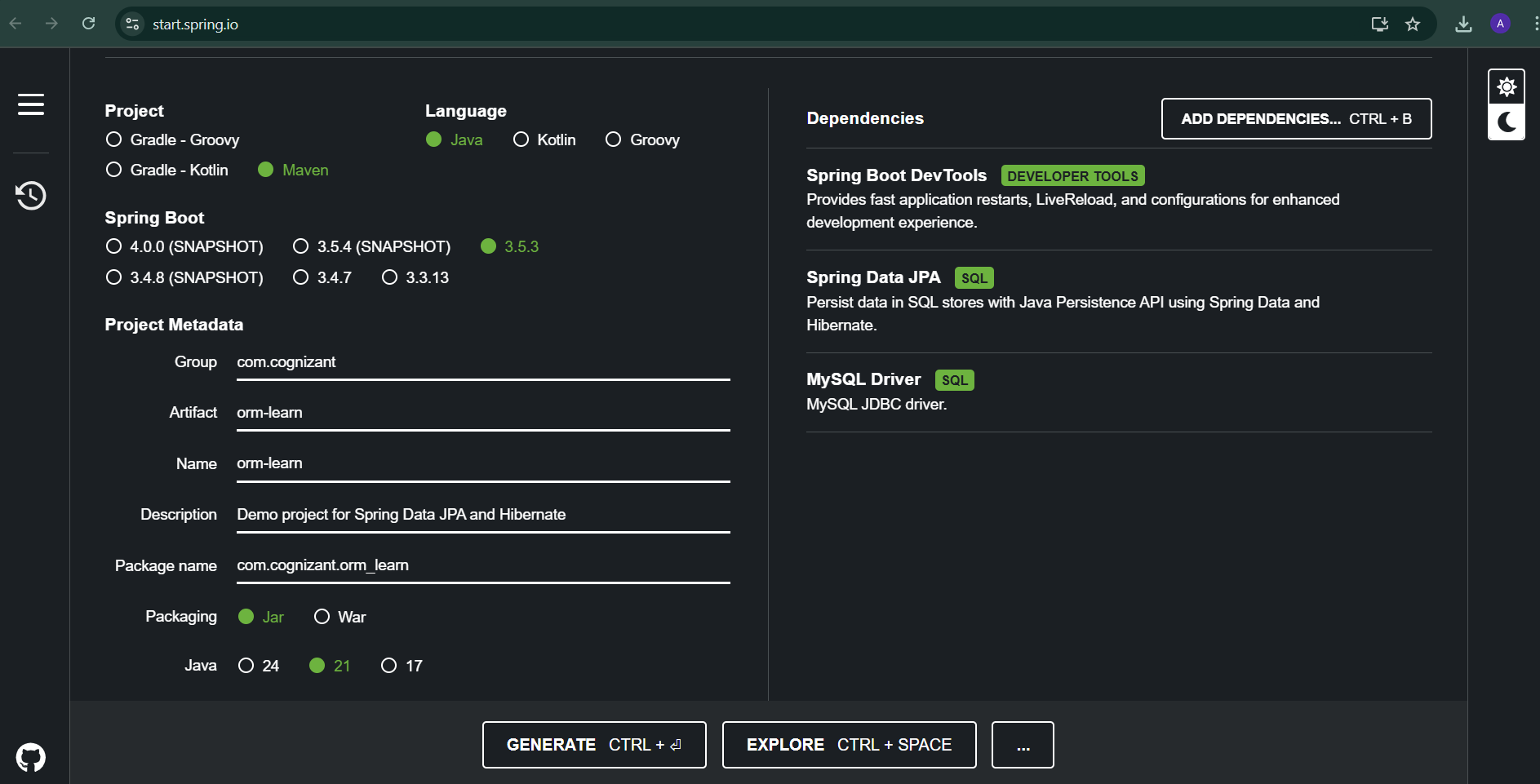
**Service Class - com.cognizant.orm-learn.service.CountryService**

**Testing in OrmLearnApplication.java**

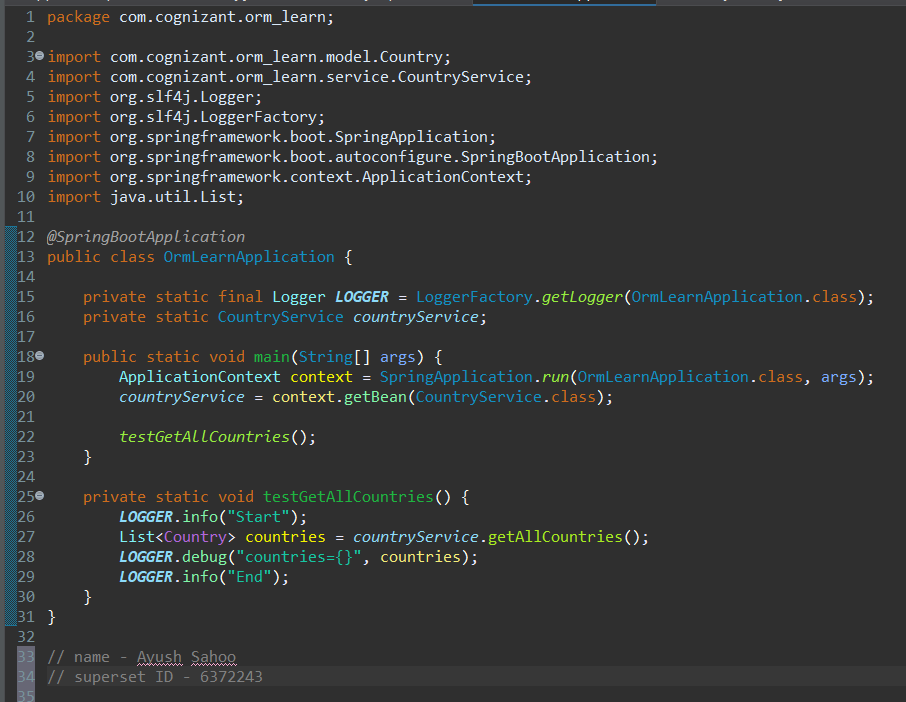
**Execute main method to check if data from ormlearn database is retrieved.**

**SOLUTION:-**

Step 1:



Step 2:

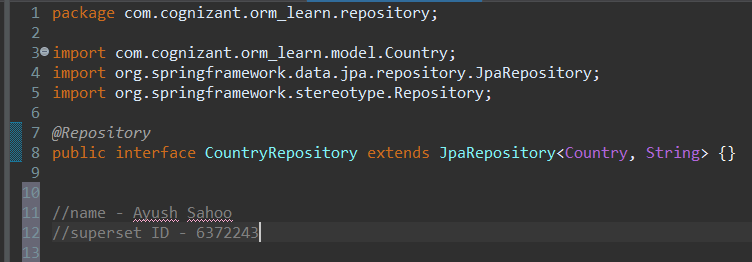


Step 3:

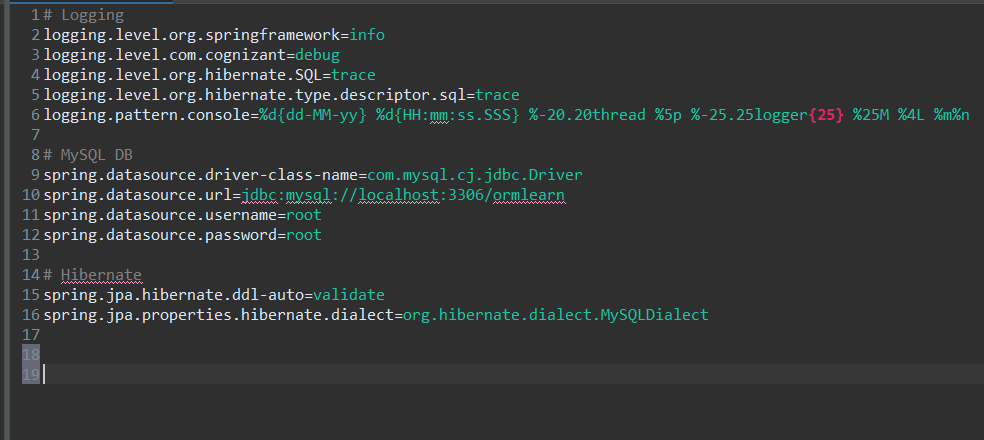
Country.java



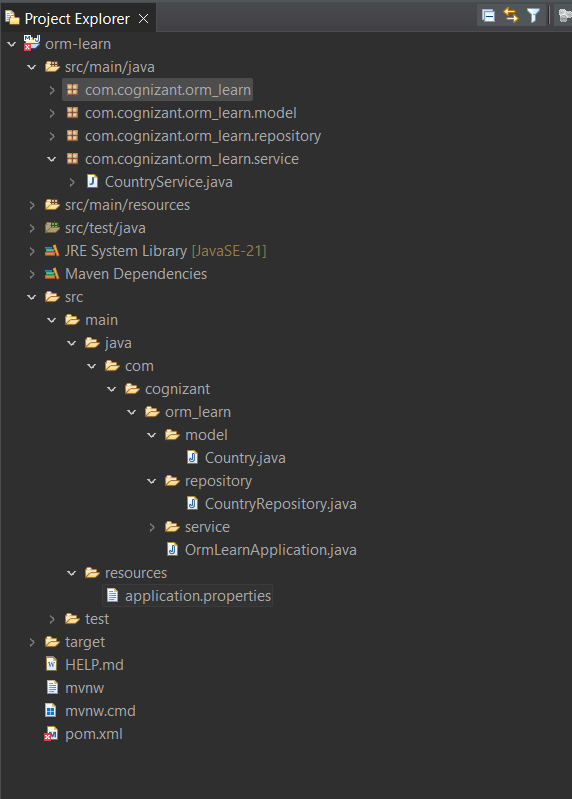
CountryRepository.java



application.properties



Below is my whole path

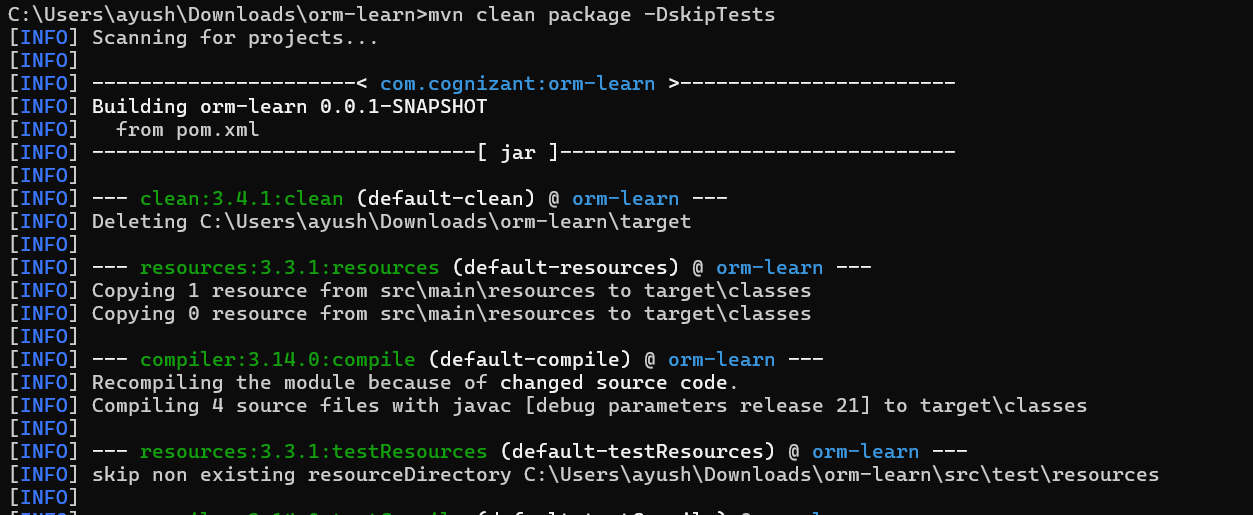


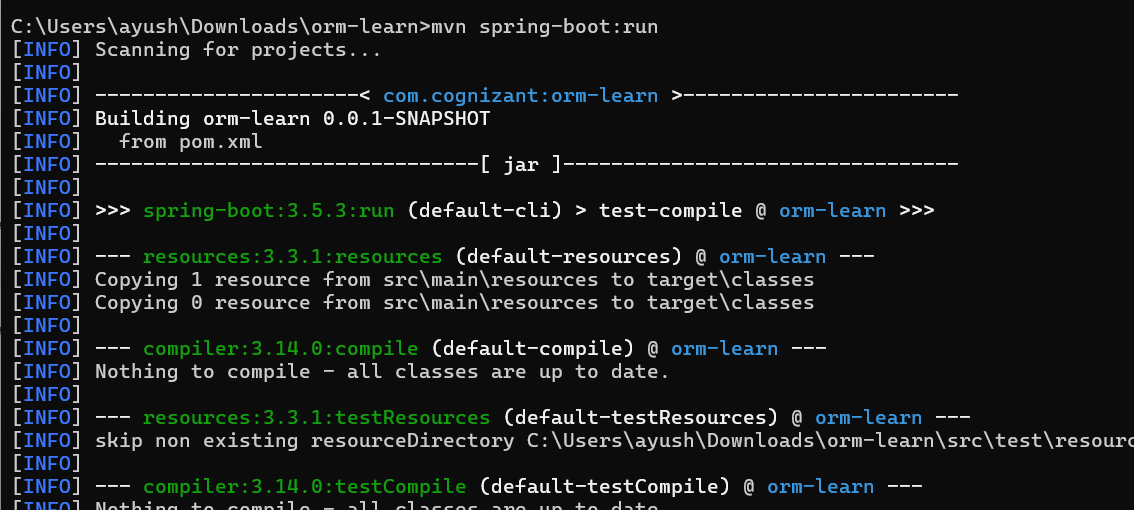
Step 4:

Commands used :

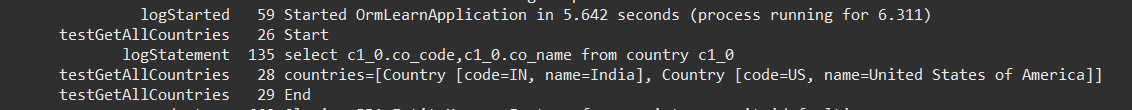
mvn clean package

spring-boot:run





**OUTPUT:**

****

**EXERCISE 4:**

**Difference between JPA, Hibernate and Spring Data JPA**

**Refer code snippets below on how the code compares between Hibernate and Spring Data JPA  
Hibernate, Spring Data JPA**

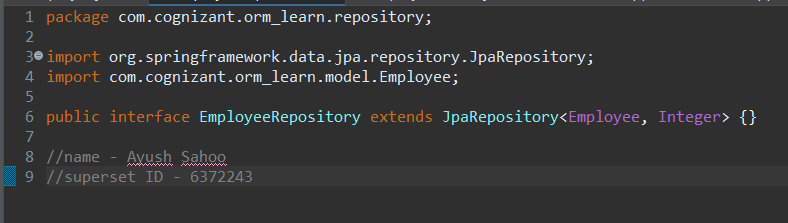
**SOLUTION:**

|  |  |  |
| --- | --- | --- |
| **Java Persistence API (JPA)** | **Hibernate** | **Spring Data JPA** |
| It is a Java Specification (JSR 338) for managing relational data in Java applications. |  It is a popular ORM (Object-Relational Mapping) tool and a JPA implementation. |  It is not a JPA provider. It sits on top of JPA and Hibernate. |
| JPA provides only the specification, not the actual implementation. | Provides extra features over JPA such as caching, lazy/eager loading, etc. | Reduces boilerplate code by providing repository interfaces. |
| You need a JPA provider (like Hibernate) to use it. | Can be used directly or as the JPA provider in a JPA-based app. | Handles automatic query generation, transaction management, and CRUD operations with minimal code. |
| It allows mapping Java objects to database tables via annotations or XML. |  | It abstracts JPA implementation (Hibernate) and integrates smoothly with Spring Boot. |

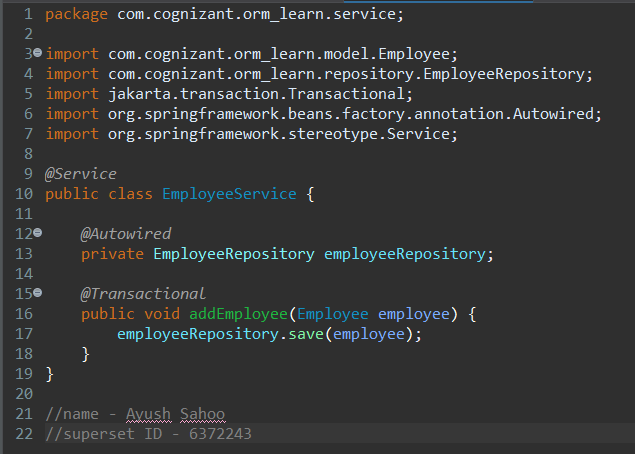
Step 1: Create “Employee.java” class under “model” folder.



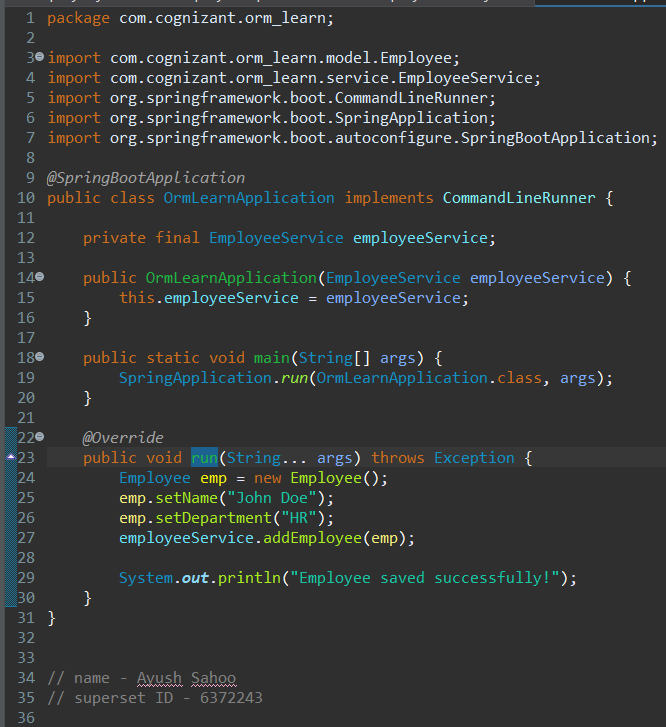
Step 2: Create “EmployeeRepository.java” class under “repository” folder.



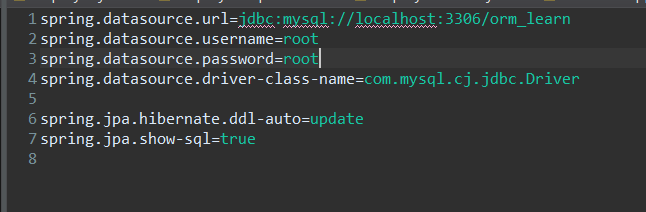
Step 3: Create “EmployeeService.java” class under “service” folder.



Step 4: Create “OrmLearnApplication.java” file.



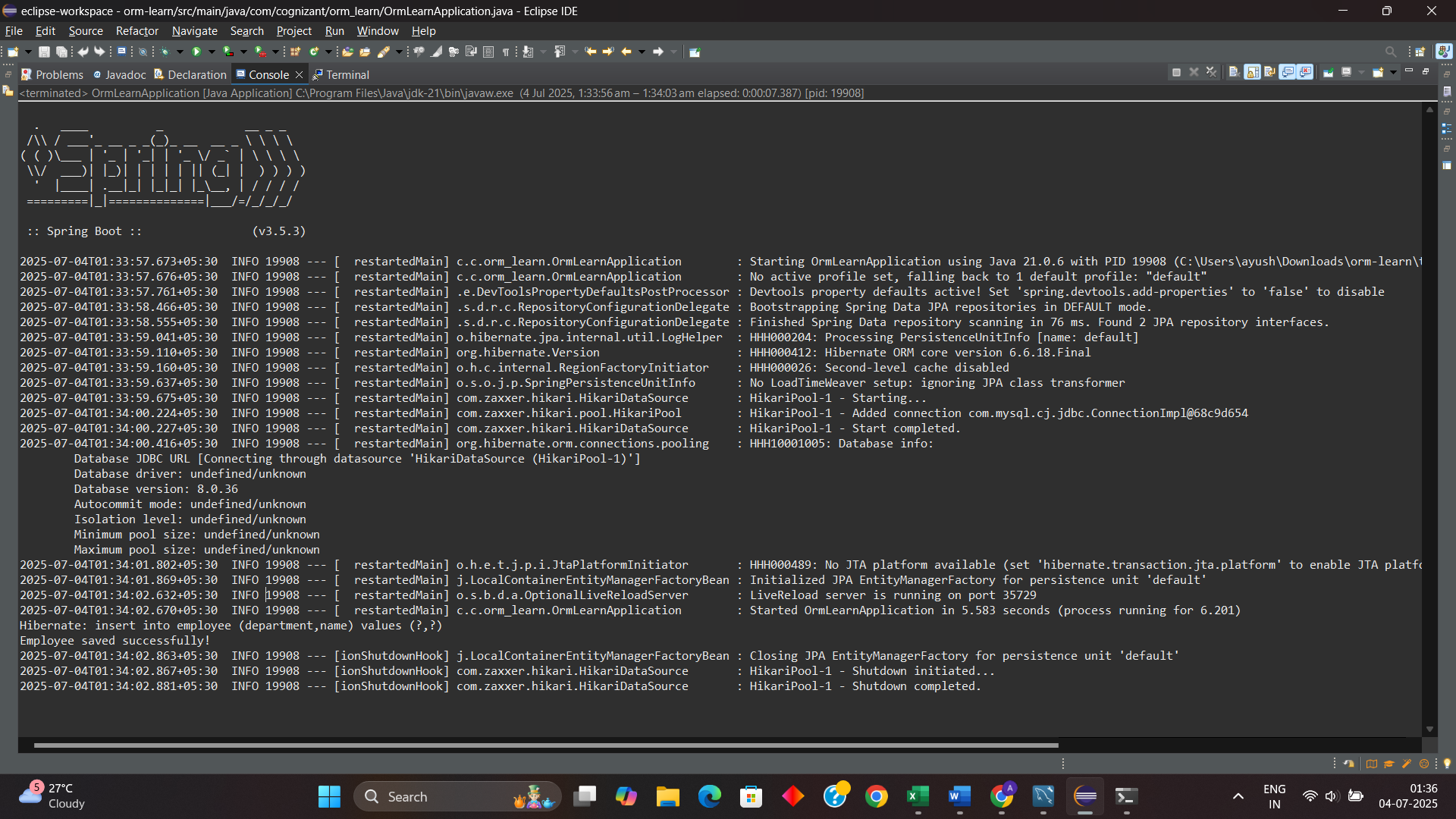
Step 5: Create “apllication.properties” file.



**OUTPUT:**

****

**Full ScreenShot:**

****

ADDITIONAL HANDS ON

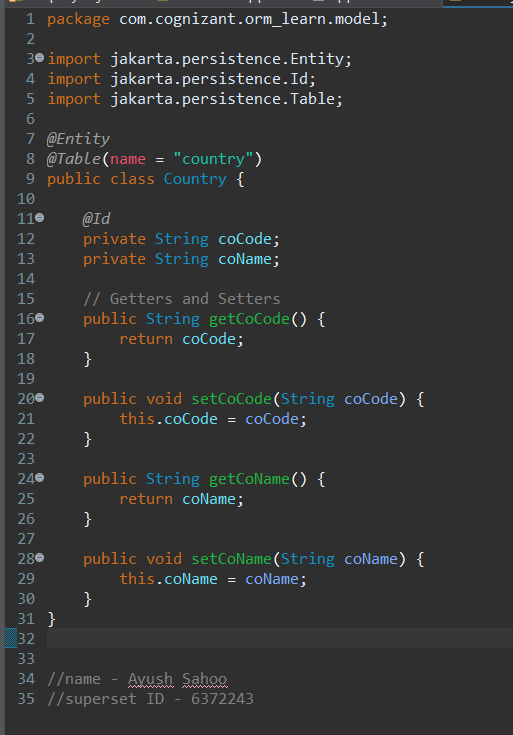
**EXERCISE 5:**

**Implement services for managing Country   
  
An application requires for features to be implemented with regards to country. These features needs to be supported by implementing them as service using Spring Data JPA.**

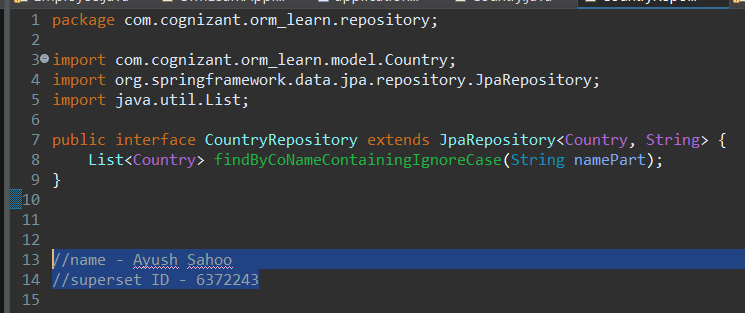
* **Find a country based on country code**
* **Add new country**
* **Update country**
* **Delete country**
* **Find list of countries matching a partial country name**

**SOLUTION:**

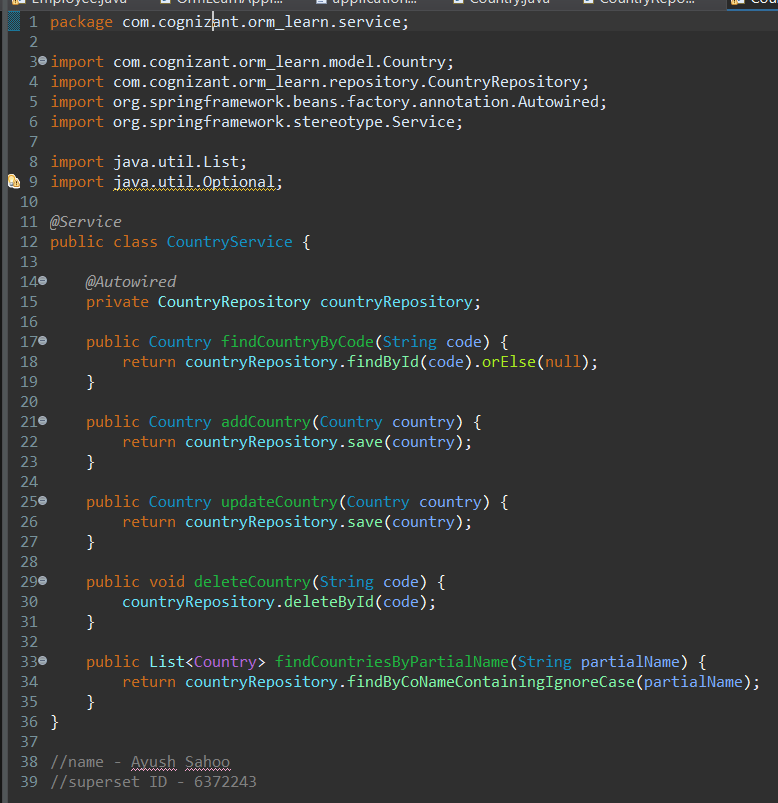
Step 1**:** Create “Country.java” file under “model” folder.

****

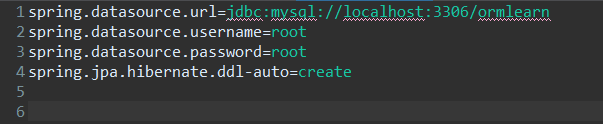
Step 2**:** Create “CountryRepository.java” file under “repository” folder.

****

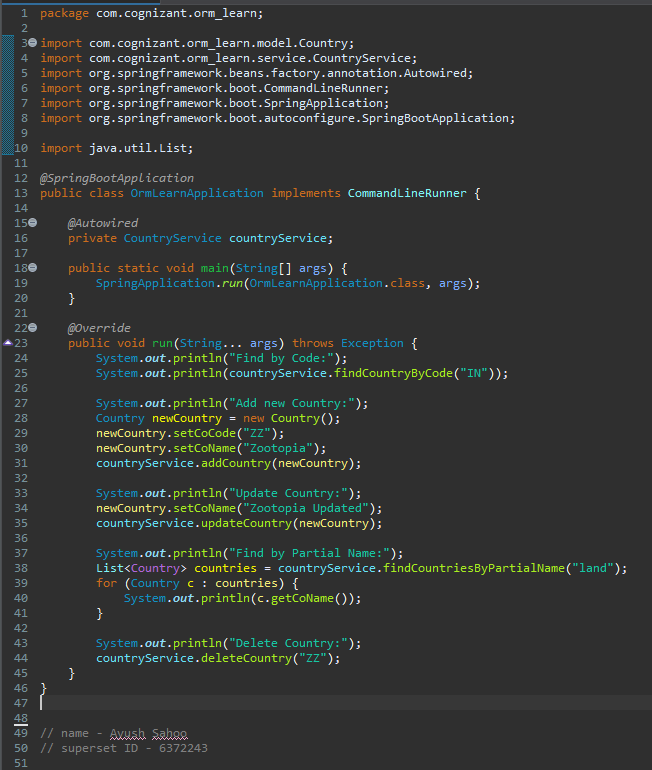
Step 3**:** Create “CountryService.java” file under “service” folder.

****

Step 4**:** Create “application.properties” file.

****

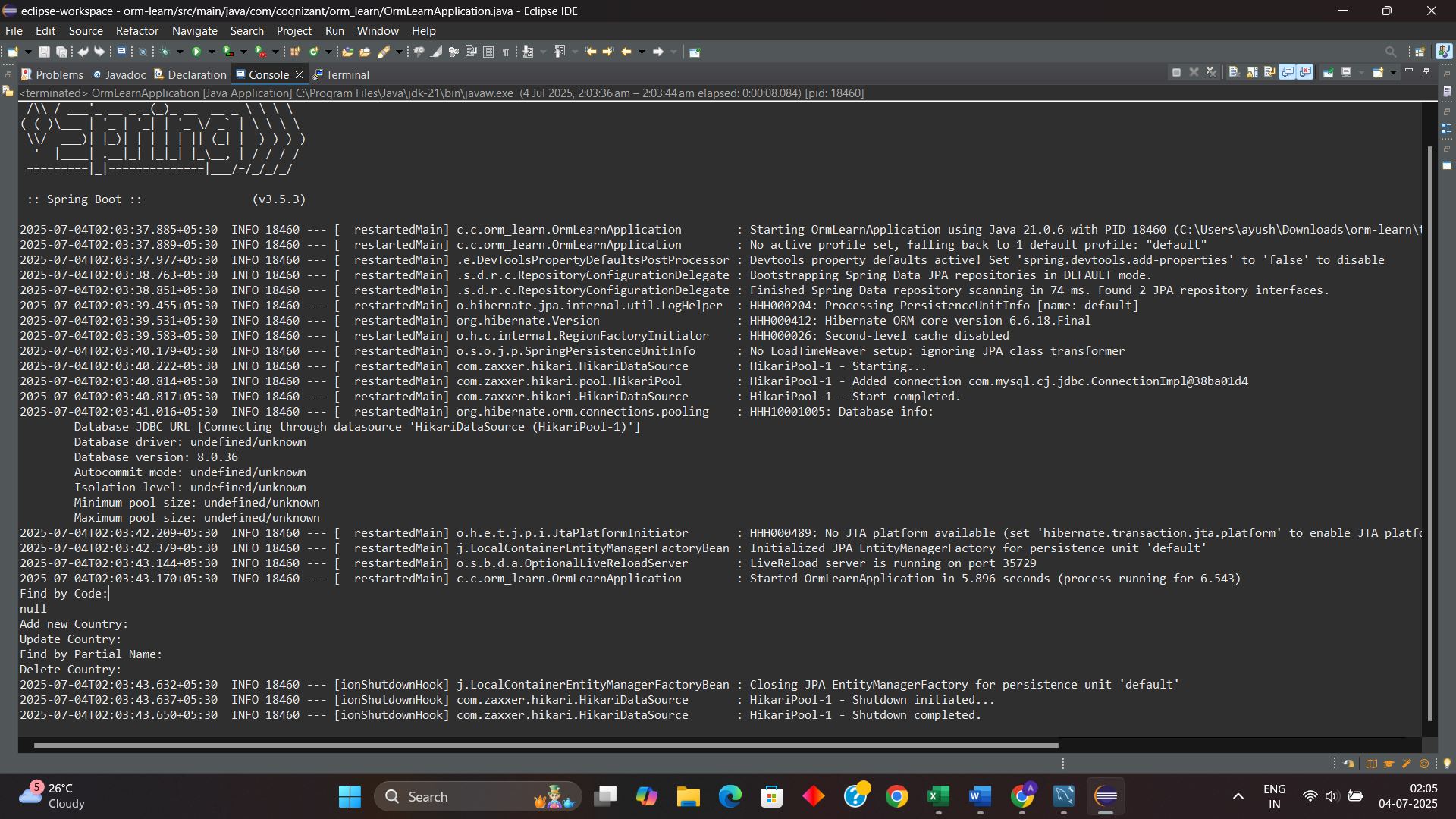
Step 4**:** Create “OrmLearnApplication.java” file.



**OUTPUT:**

****

**FULL SCREENSHOT:**

****

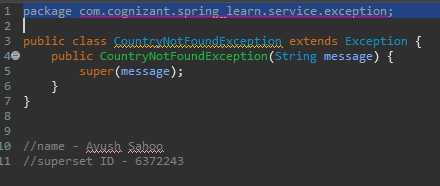
**Exercise 6**

**Find a country based on country code**

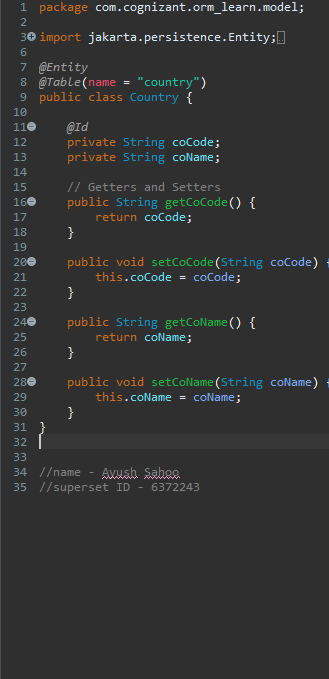
* **Create new exception class CountryNotFoundException in com.cognizant.spring-learn.service.exception**
* **Create new method findCountryByCode() in CountryService with @Transactional annotation**
* **In findCountryByCode() method, perform the following steps:**
* **Get the country based on findById() built in method**
* **From the result, check if a country is found. If not found, throw Use get() method to return the country fetched.**
* **Include new test method in OrmLearnApplication to find a country based on country code and compare the country name to check if it is valid.**
* **Invoke the above method in main() method and test it.**

**NOTE: SME to explain the importance of @Transactional annotation. Spring takes care of creating the Hibernate session and manages the transactionality when executing the service method.**

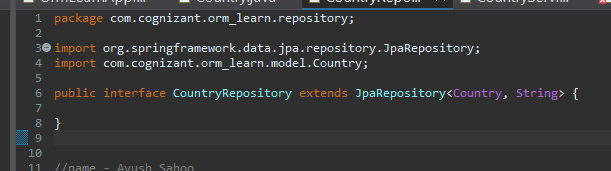
Step 1:



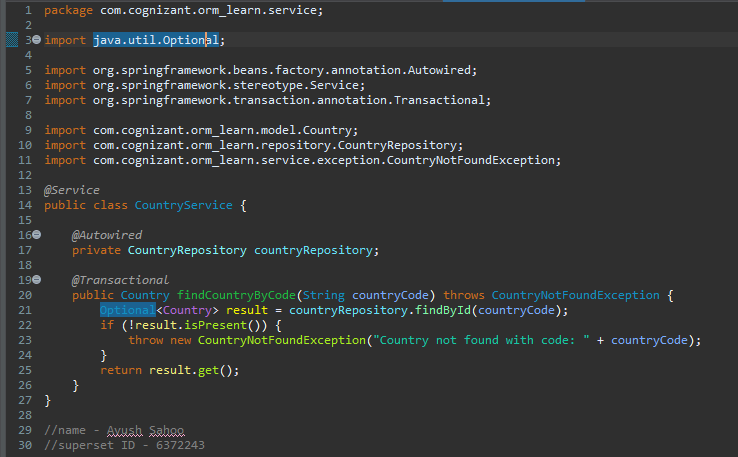
Step 2:



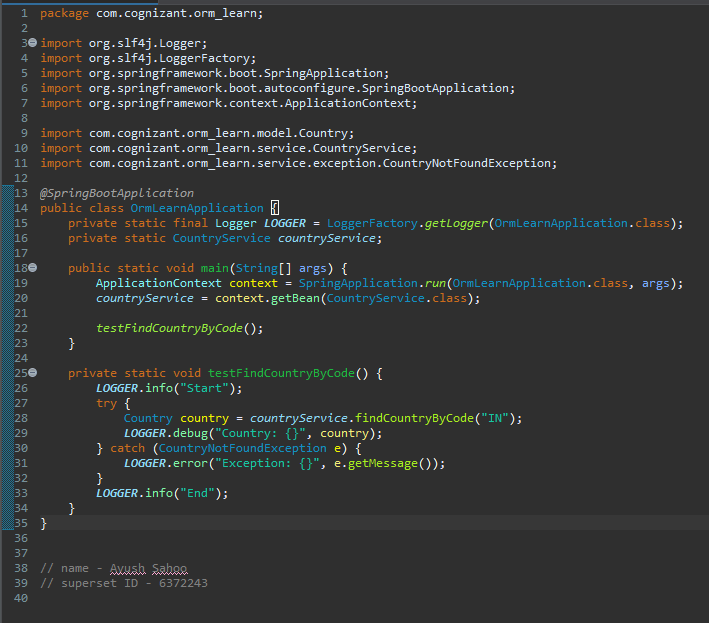
Step 3:



Step 4:



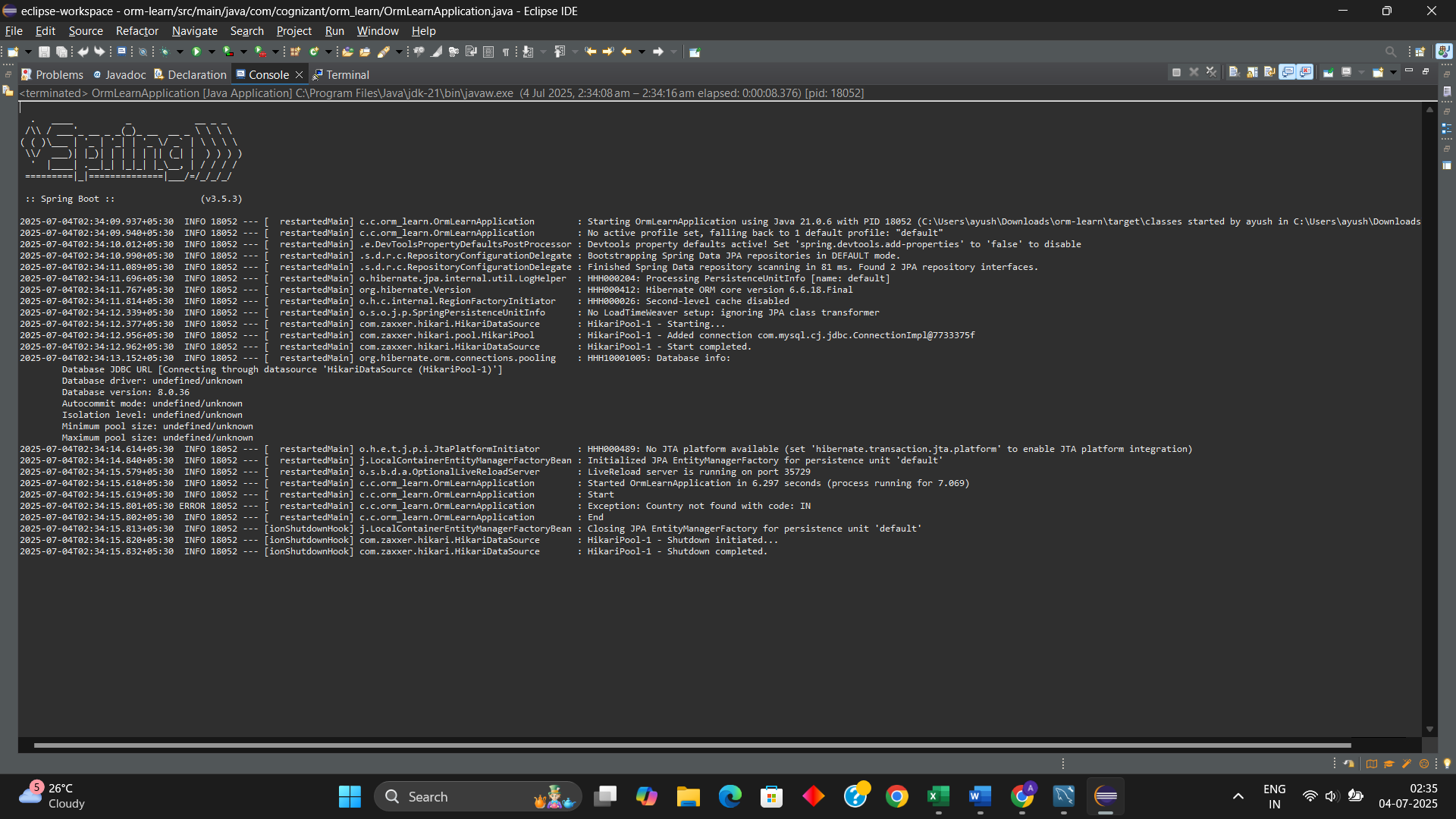
Step 5:



**OUTPUT:**

****

**FULL SCREENSHOT:**

****

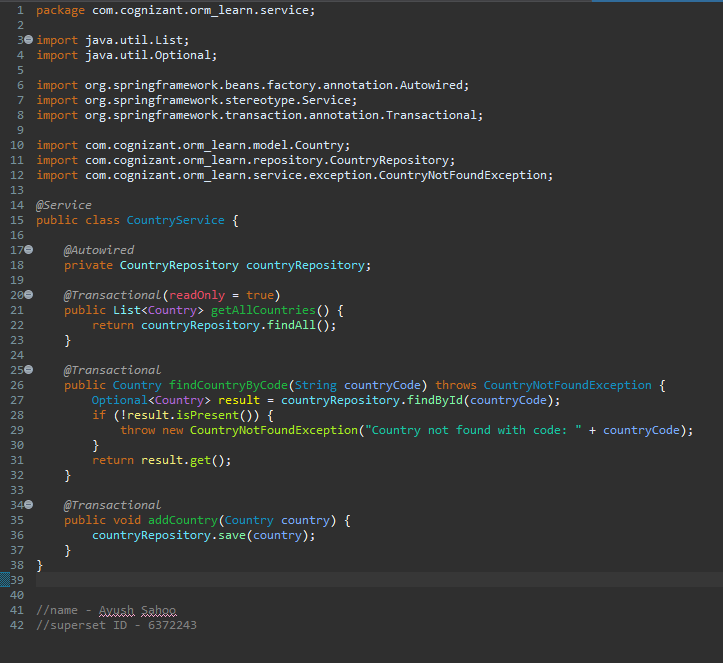
**EXERCISE 7**

**Add a new country**

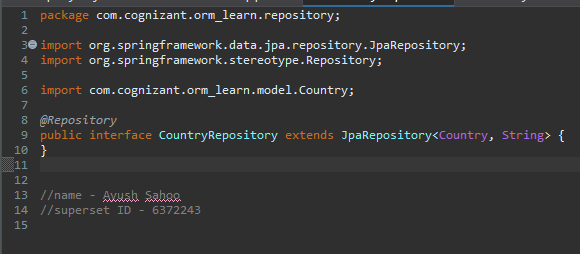
* **Create new method in CountryService.**
* **Invoke save() method of repository to get the country added.**
* **Include new testAddCountry() method in OrmLearnApplication. Perform steps below:**
  + **Create new instance of country with a new code and name**
  + **Call countryService.addCountry() passing the country created in the previous step.**
  + **Invoke countryService.findCountryByCode() passing the same code used when adding a new country**
  + **Check in the database if the country is added**

**SOLUTION:**

Step 1:



Step 2:



Step 3:

****

**OUTPUT:**





**x----x----x**