**WEEK 3 HANDS ON EXERCISE – SPRING DATA JPA WITH SPRING BOOT, HIBERNATE**

**Exercise : Spring Data JPA - Example**

**PROBLEM STATEMENT:**

Build a Spring Boot application using **Spring Data JPA** to manage student data in a single table student with fields

**CODE :**

**StudentRecordsApplication.java**

package com.Records;

import com.Records.model.Student;

import com.Records.service.StudentService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class StudentRecordsApplication implements CommandLineRunner {

*@Autowired*

private StudentService studentService;

public static void main(String[] args) {

SpringApplication.*run*(StudentRecordsApplication.class, args);

}

*@Override*

public void run(String... args) {

studentService.addStudent(new Student("Alice", "A"));

studentService.addStudent(new Student("Bob", "B"));

studentService.addStudent(new Student("Charlie", "A"));

System.***out***.println("\nAll Students:");

studentService.getAllStudents().forEach(System.***out***::println);

System.***out***.println("Student with ID 1:");

studentService.getStudentById(1).ifPresent(System.***out***::println);

System.***out***.println("Students with grade A:");

studentService.getStudentsByGrade("A").forEach(System.***out***::println);

studentService.deleteStudentById(2);

System.***out***.println("Deleted student with ID 2");

System.***out***.println("Final Student List:");

studentService.getAllStudents().forEach(System.***out***::println);

}

}

**Student.java**

package com.Records.model;

import jakarta.persistence.\*;

*@Entity*

*@Table*(name = "student")

public class Student {

*@Id*

*@GeneratedValue*(strategy = *GenerationType*.***IDENTITY***)

private Integer id;

*@Column*(name = "student\_name")

private String name;

*@Column*(name = "grade")

private String grade;

public Student() {}

public Student(String name, String grade) {

this.name = name;

this.grade = grade;

}

public Integer getId() { return id; }

public void setId(Integer id) { this.id = id; }

public String getName() { return name; }

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

public String getGrade() { return grade; }

public void setGrade(String grade) { this.grade = grade; }

*@Override*

public String toString() {

return "Student [id=" + id + ", name=" + name + ", grade=" + grade + "]";

}

}

**StudentRepository.java**

package com.Records.repository;

import com.Records.model.Student;

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

import java.util.List;

public interface StudentRepository extends JpaRepository<Student, Integer> {

List<Student> findByGrade(String grade);

}

**StudentService.java**

package com.Records.service;

import com.Records.model.Student;

import com.Records.repository.StudentRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

import java.util.Optional;

*@Service*

public class StudentService {

*@Autowired*

private StudentRepository repo;

public Student addStudent(Student student) {

return repo.save(student);

}

public List<Student> getAllStudents() {

return repo.findAll();

}

public Optional<Student> getStudentById(int id) {

return repo.findById(id);

}

public List<Student> getStudentsByGrade(String grade) {

return repo.findByGrade(grade);

}

public void deleteStudentById(int id) {

repo.deleteById(id);

}

}

**application.properties**

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.h2.console.enabled=true

**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>

<!-- ✅ Use Spring Boot parent for managed versions -->

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.2.4</version>

<relativePath/>

</parent>

<groupId>com.Records</groupId>

<artifactId>StudentRecords</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>StudentRecords</name>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

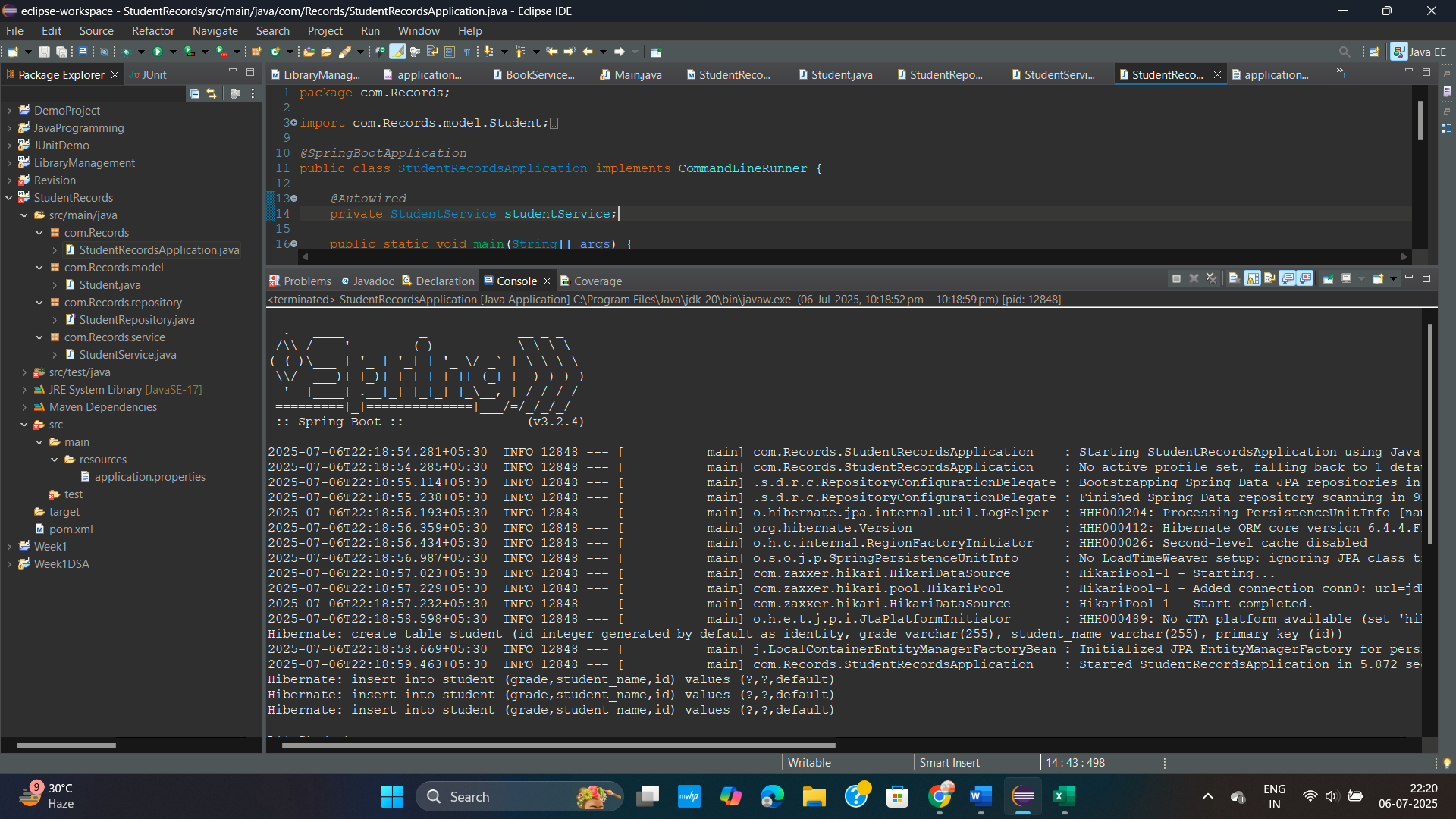
</plugin>

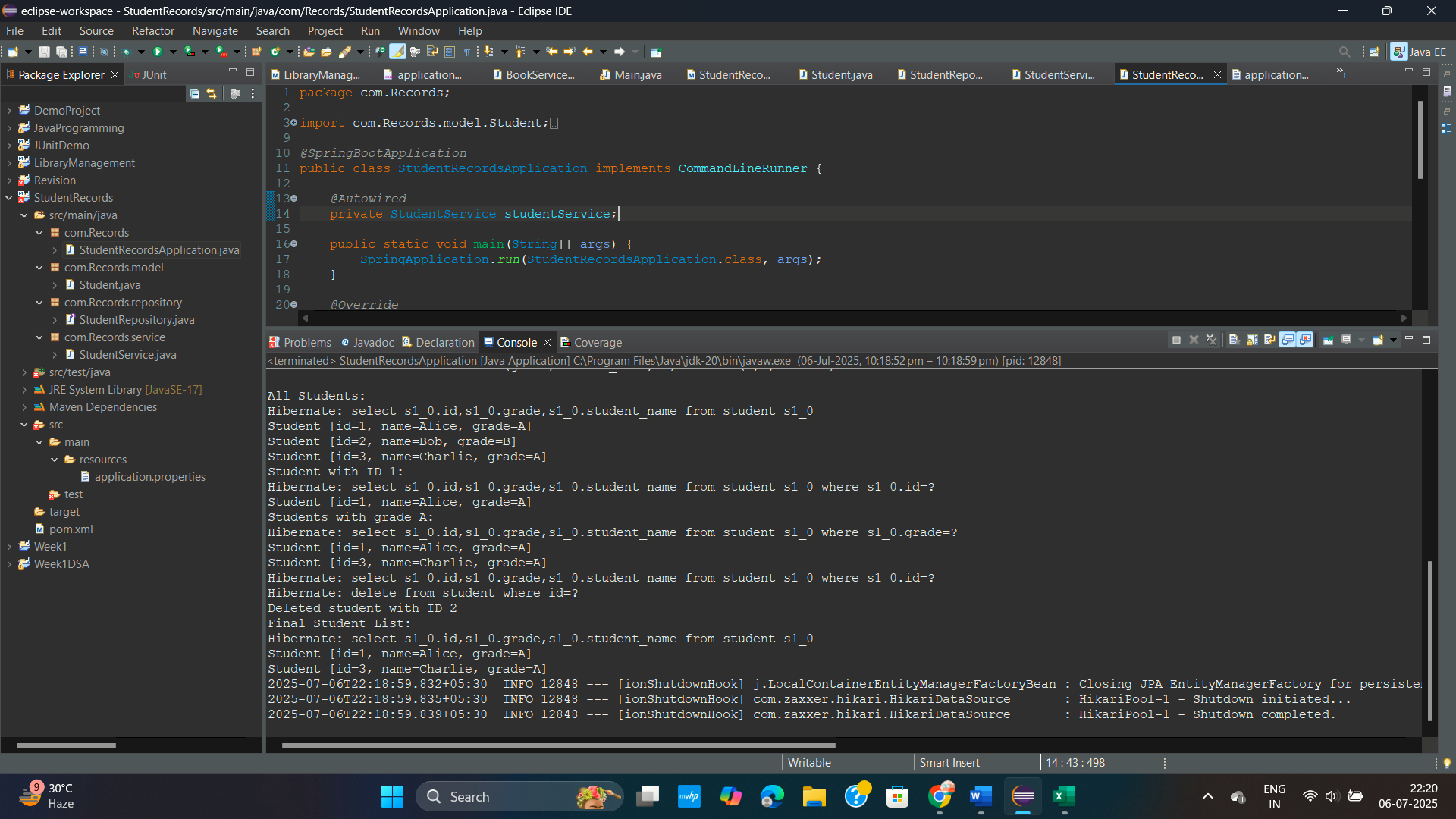
</plugins>

</build>

</project>

**OUTPUT :**





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

**PROBLEM STATEMENT :**

Create a Maven-based Java application to **store a Task** in a database using:

* **JPA** — defines the **standard** for object-relational mapping.
* **Hibernate** — a **JPA implementation** that provides ORM capabilities.
* **Spring Data JPA** — a **Spring abstraction** over Hibernate to reduce boilerplate code.

**CODE :**

**App.java**

package com.ToDo.Task;

import jakarta.persistence.\*;

*@Entity*

public class App {

*@Id*

*@GeneratedValue*

private Long id;

private String description;

private String status; // e.g., "Pending", "Completed"

// Getters and setters

public Long getId() { return id; }

public String getDescription() { return description; }

public void setDescription(String description) { this.description = description; }

public String getStatus() { return status; }

public void setStatus(String status) { this.status = status; }

}

**Main.java**

package com.ToDo.Task.Main;

import com.ToDo.Task.\*;

import jakarta.persistence.\*;

public class Main {

public static void main(String[] args) {

EntityManagerFactory emf = Persistence.*createEntityManagerFactory*("taskPU");

EntityManager em = emf.createEntityManager();

App task = new App();

task.setDescription("Complete Java Assignment");

task.setStatus("Pending");

em.getTransaction().begin();

em.persist(task);

em.getTransaction().commit();

em.close();

emf.close();

System.*setProperty*("org.jboss.logging.provider", "slf4j");

System.***out***.println("✅ Task Saved: " + task.getDescription() + " [" + task.getStatus() + "]");

}

}

**Persistence.xml**

<persistence xmlns="https://jakarta.ee/xml/ns/persistence" version="3.0">

<persistence-unit name="taskPU">

<class>com.example.model.Task</class>

<properties>

<property name="jakarta.persistence.jdbc.url" value="jdbc:h2:mem:testdb"/>

<property name="jakarta.persistence.jdbc.driver" value="org.h2.Driver"/>

<property name="jakarta.persistence.jdbc.user" value="sa"/>

<property name="hibernate.dialect" value="org.hibernate.dialect.H2Dialect"/>

<property name="hibernate.hbm2ddl.auto" value="update"/>

<property name="hibernate.show\_sql" value="true"/>

</properties>

</persistence-unit>

</persistence>

**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.example</groupId>

<artifactId>TaskApp</artifactId>

<version>1.0-SNAPSHOT</version>

<!-- ✅ Dependencies should go inside here -->

<dependencies>

<!-- JPA API -->

<dependency>

<groupId>jakarta.persistence</groupId>

<artifactId>jakarta.persistence-api</artifactId>

<version>3.1.0</version>

</dependency>

<!-- Hibernate -->

<dependency>

<groupId>org.hibernate.orm</groupId>

<artifactId>hibernate-core</artifactId>

<version>6.2.7.Final</version>

</dependency>

<!-- H2 Database -->

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<version>2.1.214</version>

</dependency>

<!-- ✅ Log4j to suppress Hibernate logs -->

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-core</artifactId>

<version>2.20.0</version>

</dependency>

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-api</artifactId>

<version>2.20.0</version>

</dependency>

</dependencies>

<!-- ✅ Compiler plugin to avoid Java 7 error -->

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>17</source>

<target>17</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

**OUTPUT :**

