**WEEK 3  
Topic 1:** Spring Core And Maven

**Exercise 1:**

Configuring a Basic Spring Application

**Code :**  
  
//AppLauncher.java

package org;

import org.exercise2.BookingApp2;

import org.exercise3.VehicleApp;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class AppLauncher {

public static void main(String[] args) {

ApplicationContext ctx = new ClassPathXmlApplicationContext("applicationContext.xml");

BookingApp2 app2 = ctx.getBean("bookingApp2", BookingApp2.class);

app2.run();

ApplicationContext ctx3 = new ClassPathXmlApplicationContext("applicationContext-ex3.xml");

VehicleApp app3 = ctx3.getBean(VehicleApp.class);

app3.run();

}

}

//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>org.example</groupId>

<artifactId>spring-full-demo</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.22</version>

</dependency>

<dependency>

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

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

<version>2.7.10</version>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<version>1.4.200</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.3.1</version>

</dependency>

<dependency>

<groupId>org.glassfish.jaxb</groupId>

<artifactId>jaxb-runtime</artifactId>

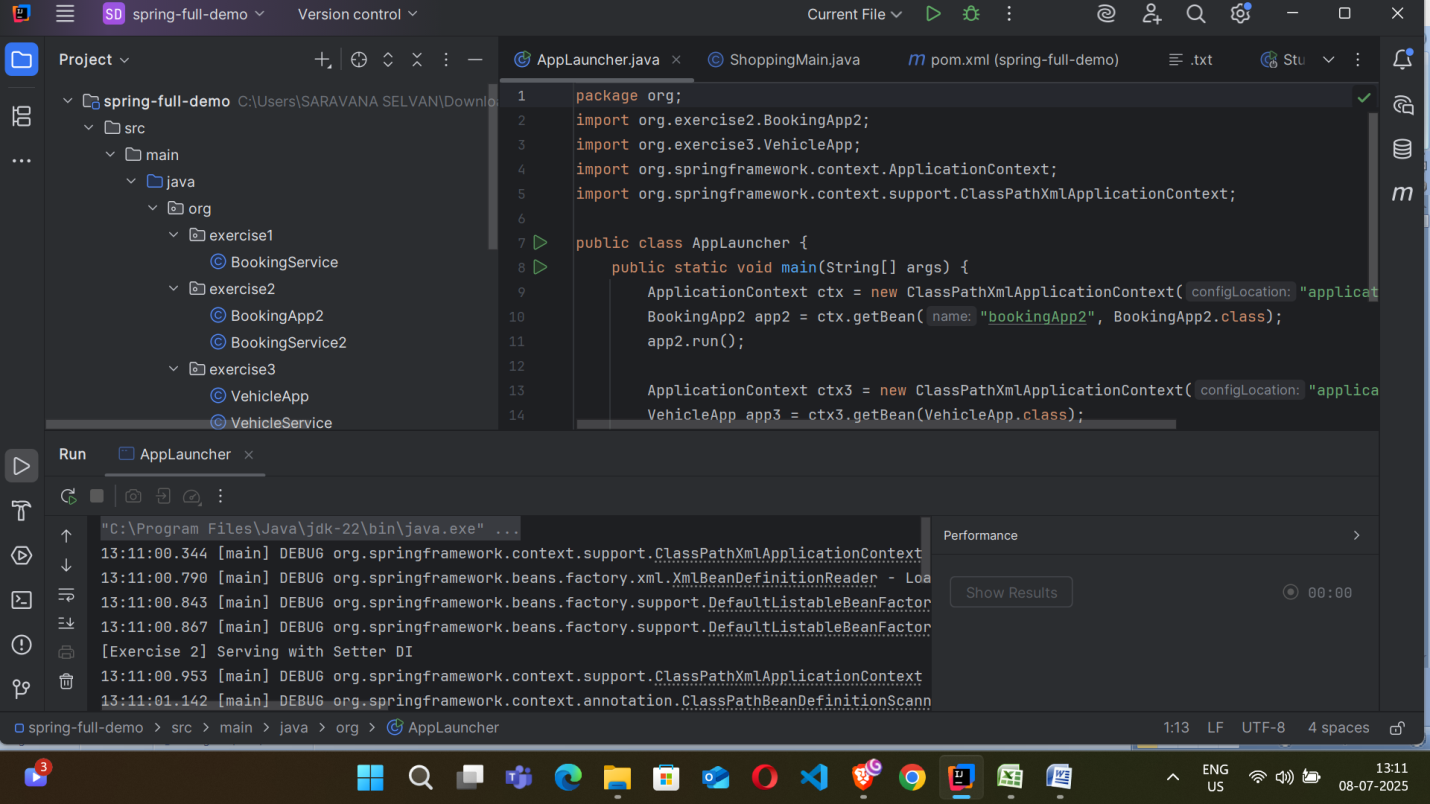
<version>2.3.1</version>

</dependency>

</dependencies>

</project>

**O/P:**



**Exercise 2:**

Implementing Dependency Injection  
  
**Code :**  
//BookingApp2.java

package org.exercise2;

public class BookingApp2 {

private BookingService2 bookingService;

public void setBookingService(BookingService2 bookingService) {

this.bookingService = bookingService;

}

public void run() {

bookingService.serve();

}

}

//BookingService2.java

package org.exercise2;

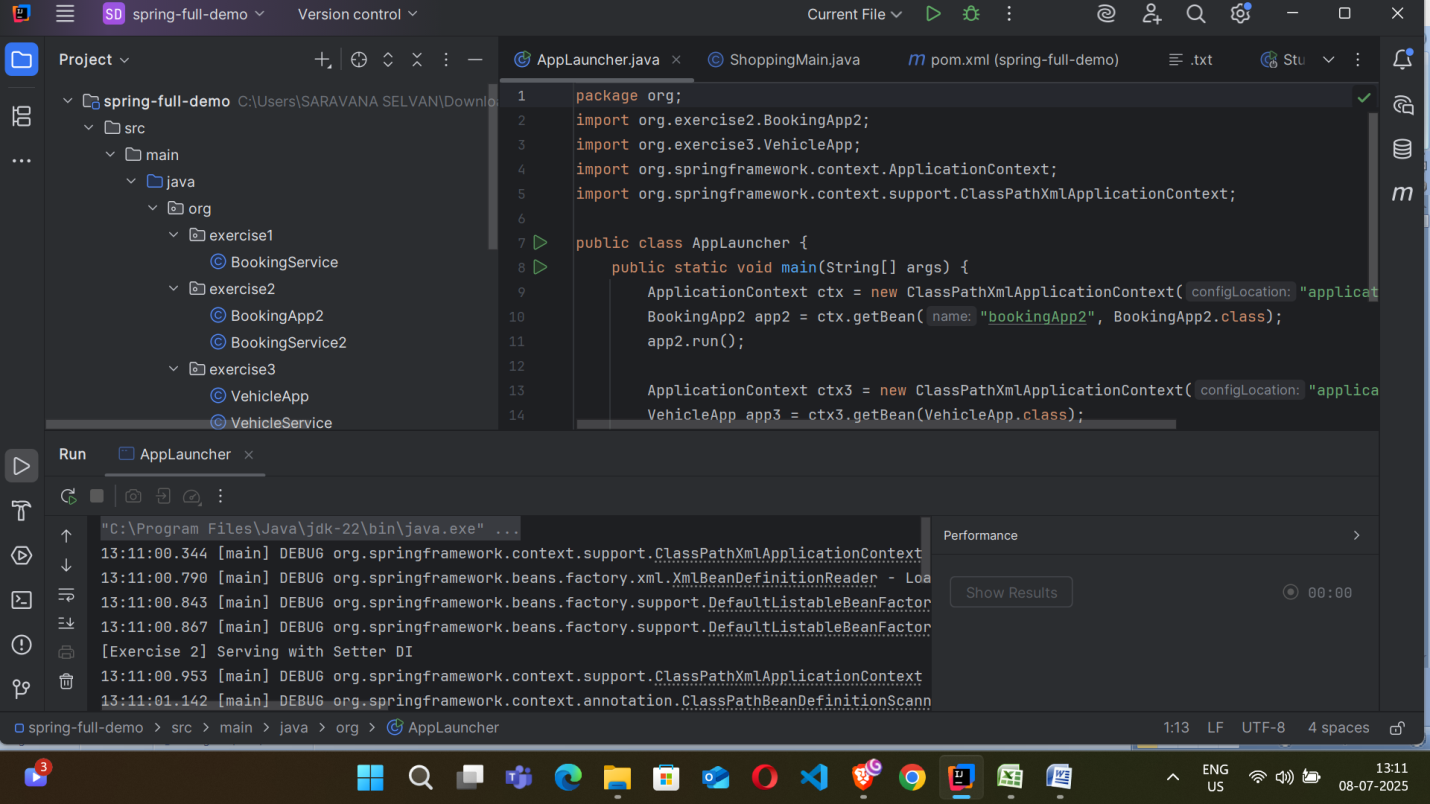
public class BookingService2 {

public void serve() {

System.out.println("[Exercise 2] Serving with Setter DI");

}

}

**O/P:**  
  


**Exercise 3:** Creating and Configuring a Maven Project  
**Code :**

**//**VehicleApp.java

package org.exercise3;

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

import org.springframework.stereotype.Component;

@Component

public class VehicleApp {

@Autowired

private VehicleService vehicleService;

public void run() {

vehicleService.repair();

}

}

//VehicleService.java

package org.exercise3;

import org.springframework.stereotype.Component;

@Component

public class VehicleService {

public void repair() {

System.out.println("[Exercise 3] Vehicle serviced via @Component");

}

}

//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>org.example</groupId>

<artifactId>spring-full-demo</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.22</version>

</dependency>

<dependency>

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

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

<version>2.7.10</version>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<version>1.4.200</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.3.1</version>

</dependency>

<dependency>

<groupId>org.glassfish.jaxb</groupId>

<artifactId>jaxb-runtime</artifactId>

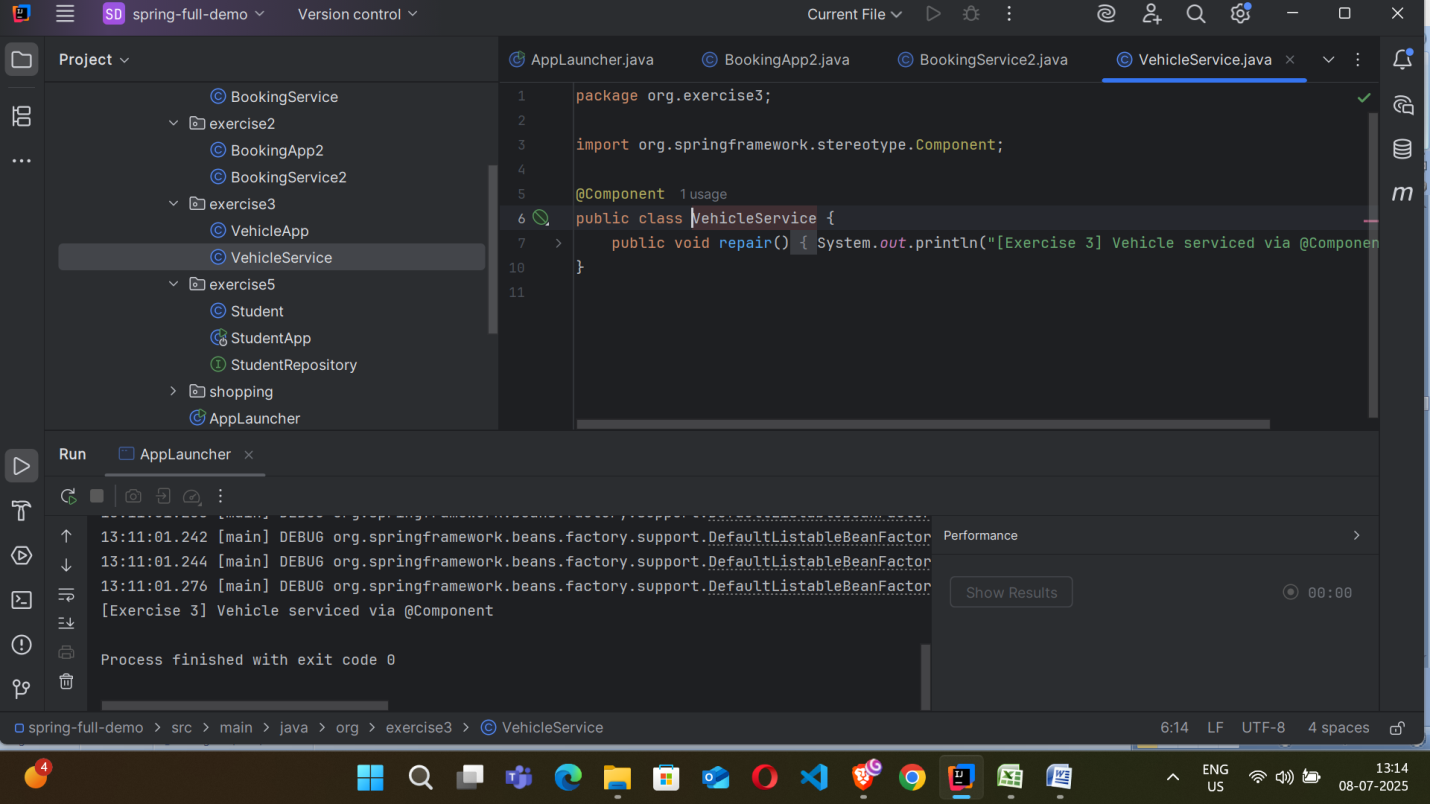
<version>2.3.1</version>

</dependency>

</dependencies>

</project>

**O/P:**

  
  
`  **Topic 2:** Spring\_Data\_JPA

**Exercise 1:**

Spring Data JPA - Quick Example  
  
**Code :**

//StudentApp.java

package org.exercise5;

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 StudentApp implements CommandLineRunner {

@Autowired

private StudentRepository repo;

public static void main(String[] args) {

SpringApplication.run(StudentApp.class, args);

}

@Override

public void run(String... args) {

repo.save(new Student("Raseeth"));

repo.save(new Student("Ansar"));

repo.findAll().forEach(s -> System.out.println(s.getName()));

}

}

//StudentRepository

package org.exercise5;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface StudentRepository extends JpaRepository<Student, Long> {}

//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>org.example</groupId>

<artifactId>spring-full-demo</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.22</version>

</dependency>

<dependency>

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

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

<version>2.7.10</version>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<version>1.4.200</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.3.1</version>

</dependency>

<dependency>

<groupId>org.glassfish.jaxb</groupId>

<artifactId>jaxb-runtime</artifactId>

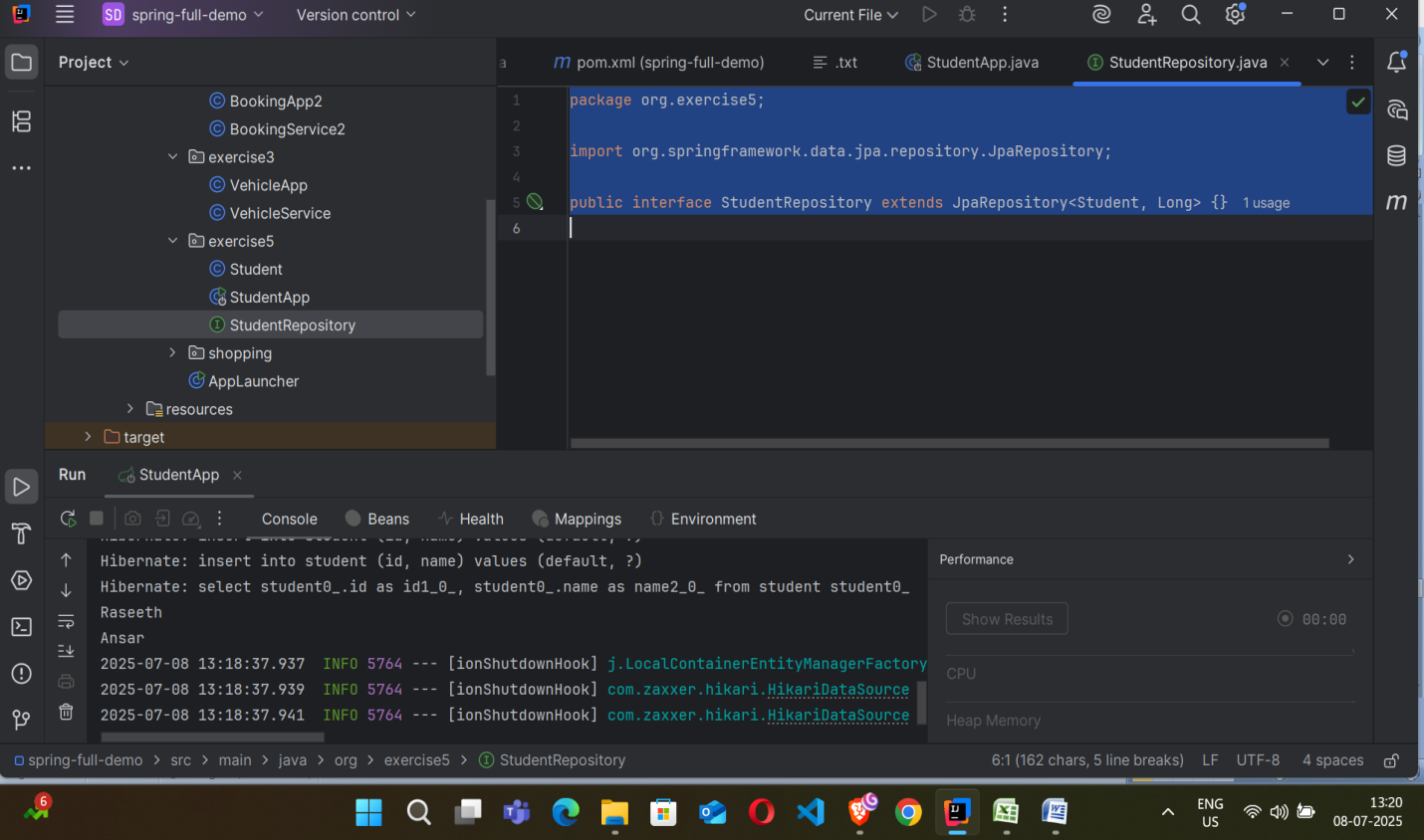
<version>2.3.1</version>

</dependency>

</dependencies>

</project>

**O/P:**

  
  
**Exercise 2:**

Difference between JPA, Hibernate and Spring Data JPA

| **Aspect** | **JPA (Java Persistence API)** | **Hibernate** | **Spring Data JPA** |
| --- | --- | --- | --- |
| **Type** | Specification (API only) | Implementation of JPA + extra features | Framework built on top of JPA and Hibernate |
| **Role** | Defines standard for Object-Relational Mapping (ORM) | Provides actual ORM implementation | Simplifies and automates JPA/Hibernate usage |
| **Persistence Logic** | None | Provided | Delegated to JPA provider (like Hibernate) |
| **Configuration** | Requires manual setup (EntityManager, transactions, etc.) | Partially reduced setup with SessionFactory and utility classes | Auto-configured in Spring Boot; minimal setup required |
| **Mapping Support** | Annotations like @Entity, @Id, @Table, etc. | Same as JPA + additional mapping features | Uses JPA annotations internally |
| **Query Language** | JPQL | JPQL, Native SQL, Criteria API | Derived query methods (e.g., findByName), JPQL, custom queries |
| **Boilerplate Code** | High – must define most logic manually | Moderate – reduces some boilerplate | Low – many operations auto-generated via interfaces |
| **Features** | Basic ORM functionality | Lazy loading, caching, dirty checking, criteria queries, more | Pagination, sorting, dynamic query creation, query DSL, automatic implementations |
| **Integration with Spring Boot** | Manual integration required | Requires additional config | Seamless integration; uses Spring’s dependency injection |
| **Ease of Use** | Moderate | Easier than plain JPA | Easiest; minimal code needed for common operations |