**Spring Data JPA – Quick Example**

pom.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>

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

    <packaging>jar</packaging>

    <name>Spring Boot Spring Data JPA</name>

    <url>https://mkyong.com</url>

    <version>1.0</version>

    <parent>

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

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

        <version>3.1.2</version>

    </parent>

    <properties>

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

        <downloadSources>true</downloadSources>

        <downloadJavadocs>true</downloadJavadocs>

    </properties>

    <dependencies>

        <!-- Spring Data JPA -->

        <dependency>

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

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

        </dependency>

        <!-- in-memory database  -->

        <dependency>

            <groupId>com.h2database</groupId>

            <artifactId>h2</artifactId>

        </dependency>

        <!-- Testing -->

        <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>

            <plugin>

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

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

                <version>3.11.0</version>

                <configuration>

                    <source>${java.version}</source>

                    <target>${java.version}</target>

                </configuration>

            </plugin>

        </plugins>

    </build>

</project>

Book.java:

package com.mkyong;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import java.math.BigDecimal;

import java.time.LocalDate;

@Entity

public class Book {

  @Id

  @GeneratedValue(strategy = GenerationType.AUTO)

  private Long id;

  private String title;

  private BigDecimal price;

  private LocalDate publishDate;

  // for JPA only, no use

  public Book() {

  }

  public Book(String title, BigDecimal price, LocalDate publishDate) {

      this.title = title;

      this.price = price;

      this.publishDate = publishDate;

  }

  //getters , setters ...

  @Override

public String toString() {

    return "Book{" +

            "id=" + id +

            ", title='" + title + '\'' +

            ", price=" + price +

            ", publishDate=" + publishDate +

            '}';

}

}

BookRepository.java:

package com.mkyong;

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

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

import org.springframework.data.repository.query.Param;

import java.time.LocalDate;

import java.util.List;

// Spring Data JPA creates CRUD implementation at runtime automatically.

public interface BookRepository extends JpaRepository<Book, Long> {

  // it works if it matches the book field name

  List<Book> findByTitle(String title);

  // Custom Query

  @Query("SELECT b FROM Book b WHERE b.publishDate > :date")

  List<Book> findByPublishedDateAfter(@Param("date") LocalDate date);

}

BookService.java:

package com.mkyong;

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

import org.springframework.stereotype.Service;

import java.time.LocalDate;

import java.util.List;

import java.util.Optional;

@Service

public class BookService {

  @Autowired

  private BookRepository bookRepository;

  public List<Book> findAll() {

      return bookRepository.findAll();

  }

  public Optional<Book> findById(Long id) {

      return bookRepository.findById(id);

  }

  public Book save(Book book) {

      return bookRepository.save(book);

  }

  public void deleteById(Long id) {

      bookRepository.deleteById(id);

  }

  public List<Book> findByPublishedDateAfter(LocalDate date) {

      return bookRepository.findByPublishedDateAfter(date);

  }

}

SpringDataJpaApplication.java:

package com.mkyong;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.Bean;

import java.math.BigDecimal;

import java.time.LocalDate;

import java.util.Optional;

@SpringBootApplication

public class SpringDataJpaApplication {

    private static final Logger log = LoggerFactory.getLogger(SpringDataJpaApplication.class);

    public static void main(String[] args) {

        SpringApplication.run(SpringDataJpaApplication.class, args);

    }

    @Bean

    public CommandLineRunner demo(BookRepository bookRepository) {

        return (args) -> {

            Book b1 = new Book("Book A", BigDecimal.valueOf(9.99), LocalDate.of(2023, 8, 31));

            Book b2 = new Book("Book B", BigDecimal.valueOf(19.99), LocalDate.of(2023, 7, 31));

            Book b3 = new Book("Book C", BigDecimal.valueOf(29.99), LocalDate.of(2023, 6, 10));

            Book b4 = new Book("Book D", BigDecimal.valueOf(39.99), LocalDate.of(2023, 5, 5));

            // Save books

            bookRepository.save(b1);

            bookRepository.save(b2);

            bookRepository.save(b3);

            bookRepository.save(b4);

            // Find all books

            log.info("findAll(), expect 4 books");

            log.info("-------------------------------");

            for (Book book : bookRepository.findAll()) {

                log.info(book.toString());

            }

            log.info("\n");

            // Find book by ID

            Optional<Book> optionalBook = bookRepository.findById(1L);

            optionalBook.ifPresent(obj -> {

                log.info("Book found with findById(1L):");

                log.info("--------------------------------");

                log.info(obj.toString());

                log.info("\n");

            });

            // Find book by title

            log.info("Book found with findByTitle('Book C')");

            log.info("--------------------------------------------");

            bookRepository.findByTitle("Book C").forEach(b -> {

                log.info(b.toString());

                log.info("\n");

            });

            // Find books published after a date

            log.info("Book found with findByPublishedDateAfter(), after 2023/7/1");

            log.info("--------------------------------------------");

            bookRepository.findByPublishedDateAfter(LocalDate.of(2023, 7, 1)).forEach(b -> {

                log.info(b.toString());

                log.info("\n");

            });

            // Delete a book

            bookRepository.deleteById(2L);

            log.info("Book deleted where ID = 2L");

            // Find all books again

            log.info("findAll() again, expect 3 books");

            log.info("-------------------------------");

            for (Book book : bookRepository.findAll()) {

                log.info(book.toString());

            }

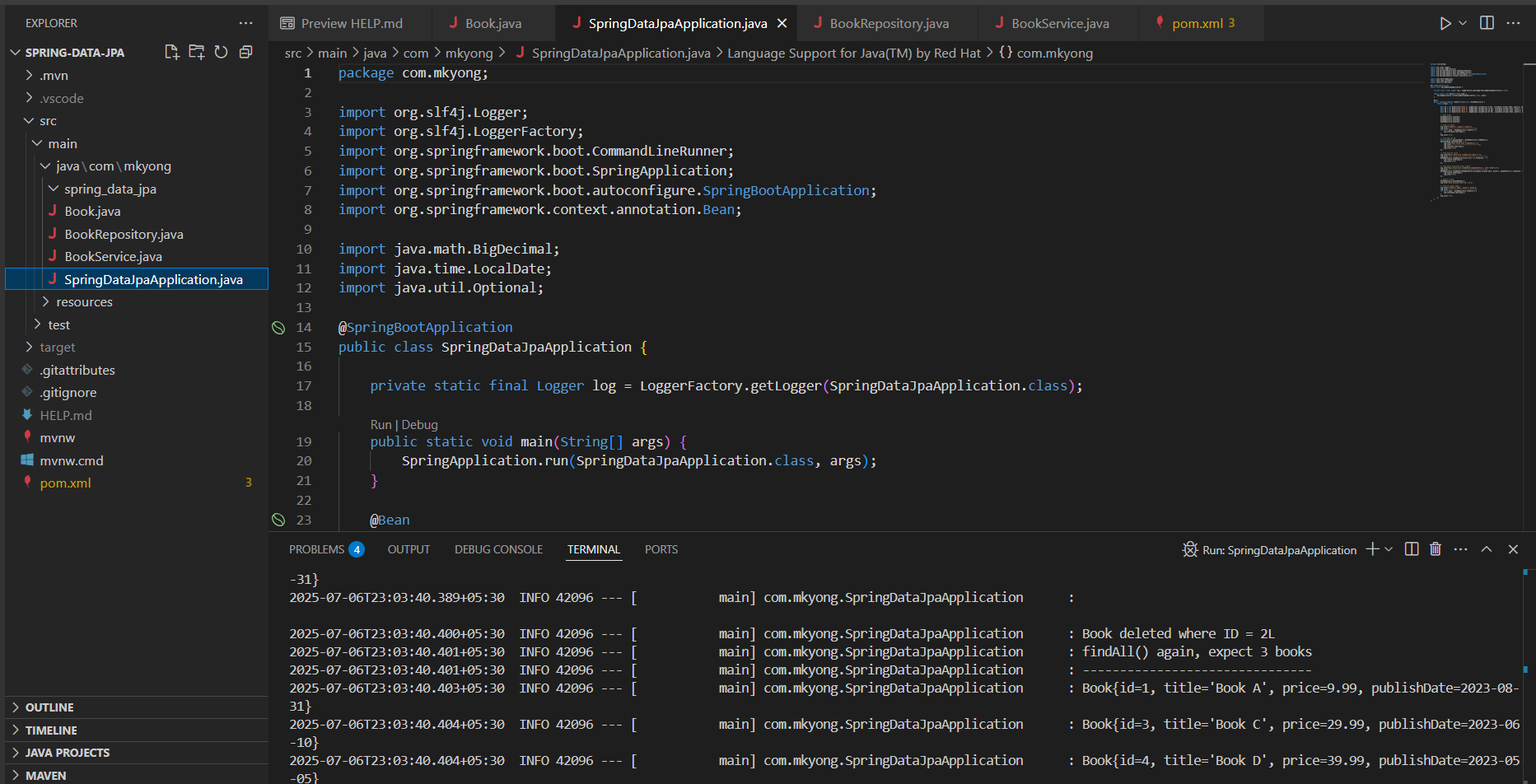
            log.info("\n");

        };

    }

}

Output:



*. \_\_\_\_ \_ \_\_ \_ \_*

*/\\ / \_\_\_'\_ \_\_ \_ \_(\_)\_ \_\_ \_\_ \_ \ \ \ \*

*( ( )\\_\_\_ | '\_ | '\_| | '\_ \/ \_` | \ \ \ \*

*\\/ \_\_\_)| |\_)| | | | | || (\_| | ) ) ) )*

*' |\_\_\_\_| .\_\_|\_| |\_|\_| |\_\\_\_, | / / / /*

*=========|\_|==============|\_\_\_/=/\_/\_/\_/*

*:: Spring Boot :: (v3.1.2)*

*2025-07-06T23:03:36.440+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Starting SpringDataJpaApplication using Java 24.0.1 with PID 42096 (C:\Users\bvssa\spring-data-jpa\spring-data-jpa\target\classes started by bvssa in C:\Users\bvssa\spring-data-jpa\spring-data-jpa)*

*2025-07-06T23:03:36.443+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : No active profile set, falling back to 1 default profile: "default"*

*2025-07-06T23:03:36.907+05:30 INFO 42096 --- [ main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.*

*2025-07-06T23:03:36.968+05:30 INFO 42096 --- [ main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 53 ms. Found 1 JPA repository interfaces.*

*2025-07-06T23:03:37.537+05:30 INFO 42096 --- [ main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...*

*2025-07-06T23:03:37.810+05:30 INFO 42096 --- [ main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection conn0: url=jdbc:h2:mem:fa17ab3b-e611-468d-b439-3450108beb45 user=SA*

*2025-07-06T23:03:37.812+05:30 INFO 42096 --- [ main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.*

*2025-07-06T23:03:37.910+05:30 INFO 42096 --- [ main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]*

*2025-07-06T23:03:38.014+05:30 INFO 42096 --- [ main] org.hibernate.Version : HHH000412: Hibernate ORM core version 6.2.6.Final*

*2025-07-06T23:03:38.019+05:30 INFO 42096 --- [ main] org.hibernate.cfg.Environment : HHH000406: Using bytecode reflection optimizer*

*2025-07-06T23:03:38.192+05:30 INFO 42096 --- [ main] o.h.b.i.BytecodeProviderInitiator : HHH000021: Bytecode provider name : bytebuddy*

*2025-07-06T23:03:38.323+05:30 INFO 42096 --- [ main] o.s.o.j.p.SpringPersistenceUnitInfo : No LoadTimeWeaver setup: ignoring JPA class transformer*

*2025-07-06T23:03:38.617+05:30 INFO 42096 --- [ main] o.h.b.i.BytecodeProviderInitiator : HHH000021: Bytecode provider name : bytebuddy*

*2025-07-06T23:03:39.283+05:30 INFO 42096 --- [ main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]*

*2025-07-06T23:03:39.309+05:30 INFO 42096 --- [ main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'*

*2025-07-06T23:03:39.698+05:30 INFO 42096 --- [ main] o.s.d.j.r.query.QueryEnhancerFactory : Hibernate is in classpath; If applicable, HQL parser will be used.*

*2025-07-06T23:03:40.218+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Started SpringDataJpaApplication in 4.249 seconds (process running for 4.686)*

*2025-07-06T23:03:40.273+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : findAll(), expect 4 books*

*2025-07-06T23:03:40.273+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : -------------------------------*

*2025-07-06T23:03:40.326+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=1, title='Book A', price=9.99, publishDate=2023-08-31}*

*2025-07-06T23:03:40.326+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=2, title='Book B', price=19.99, publishDate=2023-07-31}*

*2025-07-06T23:03:40.327+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=3, title='Book C', price=29.99, publishDate=2023-06-10}*

*2025-07-06T23:03:40.327+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=4, title='Book D', price=39.99, publishDate=2023-05-05}*

*2025-07-06T23:03:40.327+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication :*

*2025-07-06T23:03:40.336+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book found with findById(1L):*

*2025-07-06T23:03:40.336+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : --------------------------------*

*2025-07-06T23:03:40.336+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=1, title='Book A', price=9.99, publishDate=2023-08-31}*

*2025-07-06T23:03:40.336+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication :*

*2025-07-06T23:03:40.337+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book found with findByTitle('Book C')*

*2025-07-06T23:03:40.337+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : --------------------------------------------*

*2025-07-06T23:03:40.382+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=3, title='Book C', price=29.99, publishDate=2023-06-10}*

*2025-07-06T23:03:40.382+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication :*

*2025-07-06T23:03:40.383+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book found with findByPublishedDateAfter(), after 2023/7/1*

*2025-07-06T23:03:40.383+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : --------------------------------------------*

*2025-07-06T23:03:40.388+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=1, title='Book A', price=9.99, publishDate=2023-08-31}*

*2025-07-06T23:03:40.388+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication :*

*2025-07-06T23:03:40.389+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=2, title='Book B', price=19.99, publishDate=2023-07-31}*

*2025-07-06T23:03:40.389+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication :*

*2025-07-06T23:03:40.400+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book deleted where ID = 2L*

*2025-07-06T23:03:40.401+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : findAll() again, expect 3 books*

*2025-07-06T23:03:40.401+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : -------------------------------*

*2025-07-06T23:03:40.403+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=1, title='Book A', price=9.99, publishDate=2023-08-31}*

*2025-07-06T23:03:40.404+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=3, title='Book C', price=29.99, publishDate=2023-06-10}*

*2025-07-06T23:03:40.404+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication : Book{id=4, title='Book D', price=39.99, publishDate=2023-05-05}*

*2025-07-06T23:03:40.404+05:30 INFO 42096 --- [ main] com.mkyong.SpringDataJpaApplication :*

*2025-07-06T23:03:40.408+05:30 INFO 42096 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persistence unit 'default'*

*2025-07-06T23:03:40.411+05:30 INFO 42096 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown initiated...*

*2025-07-06T23:03:40.413+05:30 INFO 42096 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdo*wn completed.