**Spring Data JPA - Quick Example**

**Dependencies:**

<dependencies>

<!-- Core dependency for Spring Data JPA -->

<dependency>

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

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

</dependency>

<!-- In-memory database for this example -->

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

</dependencies>

**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

**User.java**

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

private String email;

public User() {}

public User(String name, String email) {

this.name = name;

this.email = email;

}

public Long getId() { return id; }

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

public String getName() { return name; }

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

public String getEmail() { return email; }

public void setEmail(String email) { this.email = email; }

@Override

public String toString() {

return "User{" + "id=" + id + ", name='" + name + '\'' + ", email='" + email + '\'' + '}';

}

}

### UserRepository.java

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

import org.springframework.stereotype.Repository;

import java.util.List;

import java.util.Optional;

@Repository

public interface UserRepository extends JpaRepository<User, Long> {

Optional<User> findByEmail(String email);

List<User> findByName(String name);

}

### AppRunner.java

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

import org.springframework.boot.CommandLineRunner;

import org.springframework.stereotype.Component;

@Component

public class AppRunner implements CommandLineRunner {

@Autowired

private UserRepository userRepository;

@Override

public void run(String... args) throws Exception {

System.out.println("-------> SAVING USERS...");

userRepository.save(new User("Alice", "alice@example.com"));

userRepository.save(new User("Bob", "bob@example.com"));

System.out.println("\n-------> FINDING ALL USERS...");

userRepository.findAll().forEach(System.out::println);

System.out.println("\n-------> FINDING USER BY ID (1)...");

userRepository.findById(1L).ifPresent(System.out::println);

System.out.println("\n-------> FINDING USER BY EMAIL (bob@example.com)...");

userRepository.findByEmail("bob@example.com").ifPresent(System.out::println);

}

}

**Output:**

-------> SAVING USERS...

-------> FINDING ALL USERS...

User{id=1, name='Alice', email='alice@example.com'}

User{id=2, name='Bob', email='bob@example.com'}

-------> FINDING USER BY ID (1)...

User{id=1, name='Alice', email='alice@example.com'}

-------> FINDING USER BY EMAIL (bob@example.com)...

User{id=2, name='Bob', email='bob@example.com'}