**Spring a**pplication architecture, we create three layers - Controller, Service, and DAO/Repository layer.

If we use JPA/Hibernate then write a lot of coding while implementing DAO/Repository layer.

S**pring Data JPA provides a solution to reduce a lot of boilerplate code.**

We can use Spring Data JPA to reduce the amount of boilerplate code required to implement the data access object (DAO) layer.

Spring Data JPA is not a JPA provider. It is a library/framework that adds an extra layer of abstraction on top of our JPA provider (like Hibernate).

**Spring Data JPA uses Hibernate as a default JPA provider.**

if we create DAO for other entities like**Student, Product, Order, Employee,** etc.

We are writing a lot of code and repeating the same code for other entities (only the difference is entity type and primary key fields, the rest of the code remains the same).

**Solution**

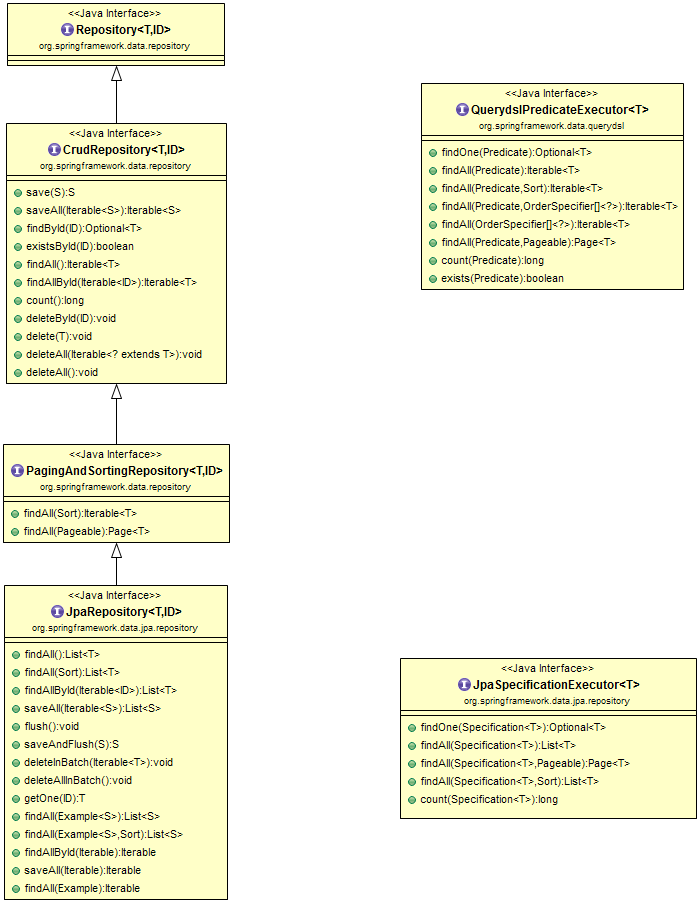
**Spring Data JPA provides a solution to reduce a lot of boilerplate code.**

Spring Data JPA provides an out-of-the-box implementation for all the required CRUD operations for the JPA entity so we don't have to write the same boilerplate code again and again.

Spring Data JPA provides repositories so we just need to extend them to get full the out-of-the-box implementation for CRUD operations for an entity.

Spring Data Commons project provides the following interfaces:

1. Repository<T, ID extends Serializable> interface
2. CrudRepository<T, ID extends Serializable> interface
3. PagingAndSortingRepository<T, ID extends Serializable> interface
4. QueryDslPredicateExecutor interface



you need to add the below dependency to your Spring boot application:

<dependency>

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

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

</dependency>

**application.properties** file and add the following configuration to it:

spring.datasource.url=jdbc:mysql://localhost:3306/demo?useSSL=false

spring.datasource.username=root

spring.datasource.password=Mysql@123

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQLDialect

spring.jpa.hibernate.ddl-auto = create-drop

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

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

public interface ProductRepository extends JpaRepository<Product, Long> {

}

<https://www.javaguides.net/p/spring-data-jpa-tutorial.html>