Name: SASHMITHA R

Superset ID: 6387562

Difference between JPA, Hibernate and Spring Data JPA

Java Persistence API (JPA)

Feature	Description
What	A specification (JSR 338) for ORM (Object Relational Mapping) in Java
Implements?	Does not provide implementation
Role	Defines how Java objects interact with relational databases (e.g., annotations like @Entity, @Id)
Use	You use JPA annotations in your entity classes. You still need a provider like Hibernate to make it work.

Example:

```
@Entity
public class Employee
{
   @Id
   private int id;
   private String name;
}
```

Hibernate

Feature	Description
What	A popular implementation of the JPA specification
Implements?	Yes
Role	Provides actual ORM behavior (SQL generation, session management, caching, etc.)

Feature	Description
Need to manage manually	Sessions, transactions, exception handling

Example (Old Hibernate-style code):

```
public Integer addEmployee(Employee employee)
{
   Session session = factory.openSession();
   Transaction tx = null;
   Integer employeeId = null;
   try {
      tx = session.beginTransaction();
      employeeId = (Integer) session.save(employee);
      tx.commit();
   } catch (HibernateException e) {
      if (tx != null) tx.rollback();
      e.printStackTrace();
   } finally {
      session.close();
   }
   return employeeId;
}
```

Spring Data JPA

Feature	Description
What	A part of Spring ecosystem that simplifies JPA usage
Implements?	Doesn't implement JPA, uses Hibernate or other providers under the hood
Role	Abstracts boilerplate code (like CRUD, sessions, transactions)
Benefit	You focus only on interface methods, Spring handles the rest

Example:

Repository Interface:

```
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {
}
```

Service Layer:

```
@Autowired
private EmployeeRepository employeeRepository;
@Transactional
public void addEmployee(Employee employee)
{
   employeeRepository.save(employee);
}
```

Comparison Summary

Aspect	**JPA**	**Hibernate**	**Spring Data JPA**
Type	Specification	Implementation of JPA	Abstraction over JPA
Provides Implementation?	No	Yes	No
Requires Provider?	Yes	No(is a provider)	Uses Hibernate, EclipseLink, etc..
Ease of Use	Moderate	Verbose	Very easy
Code Boilerplate	Some	High	Minimal
Transaction Handling	Manual	Manual	Automatic via @Transactional

---

Project Structure Overview

1. JPA + Hibernate (Plain Java SE with JPA annotations and Hibernate implementation)

- Pure Java project using Hibernate as the JPA provider
- Manual session and transaction handling

2. Hibernate Only (No JPA annotations)

- Uses Hibernate-specific configuration and mapping XML
- Manually manages Hibernate sessions and transactions

3. Spring Boot + Spring Data JPA

- Uses Spring Boot with Spring Data JPA
- Uses JpaRepository to auto-generate CRUD logic

Approach	Key Feature	Boilerplate	Notes
1. JPA + Hibernate	Uses annotations	Medium	No Spring
2. Hibernate Only	XML mapping	High	Manual config
3. Spring Data JPA	Uses Spring Boot and JpaRepository	Lowest	Most efficient