**Hands-on 4: Difference between JPA, Hibernate, and Spring Data JPA**

**Java Persistence API (JPA)**

JPA is part of the Java specification (JSR 338) for persisting, reading, and managing data from Java objects.

It defines a standard set of interfaces and annotations for Object-Relational Mapping (ORM).

JPA does not provide a concrete implementation — it only defines the specification.

Example of JPA implementations: Hibernate, EclipseLink, OpenJPA.

**Hibernate**

Hibernate is a popular ORM framework that implements the JPA specification.

It provides the actual logic to map Java objects to database tables.

Hibernate can also offer features beyond JPA (e.g., caching, native SQL, criteria queries).

**Spring Data JPA**

Spring Data JPA is an abstraction over JPA and its implementations (e.g., Hibernate).

It helps reduce boilerplate code (like writing queries, handling transactions, etc.).

It integrates tightly with Spring’s ecosystem and automatically manages transactions.

Spring Data JPA provides interfaces (like JpaRepository) with built-in CRUD and query methods.

**Comparison table**

| Aspect | JPA | Hibernate | Spring Data JPA |
| --- | --- | --- | --- |
| Type | Specification (API / interface) | ORM tool (implements JPA) | Abstraction over JPA implementation |
| Contains Implementation? | No | Yes | No (relies on Hibernate / JPA impl) |
| Purpose | Defines ORM mapping standard | Provides ORM functionality | Simplifies data access, reduces code |
| Manages transactions? | No | No (manual or with Spring) | Yes (via Spring) |
| Boilerplate code? | Depends on implementation | More code (Session, Transaction) | Minimal (uses repository interfaces) |

**Code comparison**

**Hibernate (manual)**

/\* Method to CREATE an employee in the database \*/

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**

// EmployeeRepository.java

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

// EmployeeService.java

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}

}