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

## 1. Overview

This document explains the difference between Java Persistence API (JPA), Hibernate, and Spring Data JPA in the context of Java applications using data persistence and object-relational mapping (ORM).

## 2. Java Persistence API (JPA)

- JPA is a specification defined by JSR 338 for object-relational mapping in Java.  
- It provides a standard for persisting, reading, and managing data from Java objects to relational databases.  
- JPA does not provide any concrete implementation.  
- Popular JPA implementations include Hibernate, EclipseLink, and OpenJPA.

## 3. Hibernate

- Hibernate is an ORM (Object-Relational Mapping) framework and one of the most popular implementations of JPA.  
- It simplifies database interactions by mapping Java objects to database tables.  
- Developers interact with Hibernate APIs to persist and retrieve data using Session and Transaction management.

## 4. Spring Data JPA

- Spring Data JPA is a part of the Spring Framework that provides an abstraction layer over JPA implementations like Hibernate.  
- It does not provide its own JPA implementation.  
- It significantly reduces boilerplate code by auto-generating common CRUD operations.  
- It simplifies transaction management and data access layers using annotations like @Repository and @Transactional.

## 5. Code Comparison

### Hibernate:

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

## 6. Summary

- JPA is a specification, not an implementation.  
- Hibernate is a JPA-compliant ORM framework.  
- Spring Data JPA is an abstraction layer on top of JPA implementations like Hibernate, reducing development time and effort.  
- Spring Data JPA is preferred for simplifying CRUD operations and minimizing boilerplate code in Spring applications.