Exercise 4

**JPA (Java Persistence API)**

JPA is **just an interface**; it does **not contain any implementation**.

It standardizes how Java objects should be persisted to relational databases.

Needs a **provider** (like Hibernate, EclipseLink) to work.

**Hibernate**

Hibernate is a **popular ORM framework** and the most widely used **implementation of JPA**.

Implements both **JPA specification** and offers its own **proprietary features** (like caching, HQL, interceptors).

You can use **Hibernate API directly** or **use it through JPA**.

**Spring Data JPA**

A **Spring-based abstraction** built **on top of JPA and Hibernate** (or any other JPA provider).

Not a JPA provider itself.

Helps eliminate boilerplate code (like DAO implementation).

Provides **built-in CRUD**, **pagination**, **custom queries**, **derived queries**, et

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;

}

//repository

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

//service

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}