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

JPA (Java Persistance API)

- It is the standard **specification** for working with data in Java.
- JPA is defined by JSR 338.
- It does **not provide any implementation** itself, it is just an interface level.
- It allows you to map Java classes to database tables (ORM).
- Hibernate is one of the most common implimentations of JPA.

Hibernate

- Hibernate is an **ORM tool** and it implements JPA.
- It does things like mapping class to table, automatic SQL generation, transaction handling etc.
- You need to **manually manage sessions and transactions** in Hibernate.

Spring Data JPA

- It is part of Spring framework.
- It does **not implement JPA** but it provides an abstraction layer over JPA implementation like Hibernate.
- It helps reduce **boilerplate code**.
- It auto handles CRUD methods like `save()`, `findAll()` etc.

- It handles **transactions** automatically using `@Transactional`.

Code Example Diffrence

Hibernate Code Example

```
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 Example

EmployeeRepository.java

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

EmployeeService.java

```
@Autowired
private EmployeeRepository employeeRepository;

@Transactional
public void addEmployee(Employee employee) {
```

```
    employeeRepository.save(employee);
}
```