Week3

(Difference Between JPA, Hibernate, and Spring Data JPA)

# 1. JPA (Java Persistence API)

- JPA is a standard specification in Java for managing relational data in Java applications.  
- It outlines APIs and annotations like @Entity, @Id, and EntityManager to define how Java classes should interact with database tables.  
- JPA itself doesn’t include an implementation — it only provides guidelines for object-relational mapping.  
- Frameworks like Hibernate, EclipseLink, and OpenJPA act as implementations of JPA.

# 2. Hibernate

- Hibernate is a powerful ORM (Object-Relational Mapping) tool and a widely used JPA implementation.  
- It handles the conversion between Java objects and database rows using mapping rules.  
- Beyond standard JPA features, Hibernate offers additional capabilities like second-level caching, lazy/eager loading, and custom queries.  
- When used without Spring, developers must manage transactions and sessions manually.

# 3. Spring Data JPA

- Spring Data JPA is part of the Spring ecosystem and serves as a wrapper around JPA to simplify database operations.  
- It doesn’t implement JPA directly but works with existing implementations like Hibernate.  
- It removes boilerplate code by allowing developers to create interfaces for repositories rather than manually writing queries.  
- It seamlessly integrates with Spring Boot and automatically manages transactions, reducing setup complexity.

# Code Style Comparison

## Using Hibernate (Manual Setup)

public Integer saveEmployee(Employee employee) {  
 Session session = sessionFactory.openSession();  
 Transaction transaction = null;  
 Integer empId = null;  
  
 try {  
 transaction = session.beginTransaction();  
 empId = (Integer) session.save(employee);  
 transaction.commit();  
 } catch (Exception e) {  
 if (transaction != null) transaction.rollback();  
 } finally {  
 session.close();  
 }  
  
 return empId;  
}

## Using Spring Data JPA (Simplified with Repositories)

### EmployeeRepository.java

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

### EmployeeService.java

@Service  
public class EmployeeService {  
  
 @Autowired  
 private EmployeeRepository employeeRepository;  
  
 @Transactional  
 public void saveEmployee(Employee employee) {  
 employeeRepository.save(employee);  
 }  
}