**Hands on 4**

**Difference between JPA, Hibernate and Spring Data JPA** 

**Java Persistence API (JPA)**

* JPA is a **specification (JSR 338)** for persisting, reading, and managing data using Java objects.
* It does **not provide any implementation**.
* It defines annotations like @Entity, @Id, @Table etc.
* Requires an implementation like Hibernate.

**Hibernate**

* Hibernate is an **implementation of the JPA specification**.
* It is an **ORM (Object Relational Mapping)** tool that maps Java classes to database tables.
* Provides features like caching, lazy loading, criteria API.
* Developers must **manually manage** session and transaction.

**Spring Data JPA**

* Spring Data JPA is a **Spring framework abstraction** over JPA.
* It uses Hibernate internally but removes boilerplate code.
* Provides repository interfaces like JpaRepository, CrudRepository.
* Automatically handles session and transaction management.

Code Comparison:

Hibernate Approach:

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 Approach

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

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}

}