Hibernate Approach

/\* Method to CREATE an employee in the database \*/

   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

*// Repository interface*

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

*// Service class*

@Transactional

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}

|  |  |
| --- | --- |
| Hibernate Approach | Spring Data JPA Approach |
| * Manual Session management * openSession() * close() * Explicit transaction handling * beginTransaction() * commit() * rollback()      * Boilerplate code * Direct Hibernate API usage * Checked exceptions | * No Session Management * Declarative Transactions * @Transactions * Minimal boilerplate * Repository Abstraction * Runtime Exceptions |