Spring orm theory

There is pain with spring jdbc to manually create table.
Through arm we can map a class directly with table.

For arm we can use jpa.
Actual implementation of Jpa is by hibernate.

So we have a chain like

Spring App -> spring orm -> spring hibernate -> database.

Instance creation for hibernate will be done by spring orm, configuration by spring configuration.
Transactions will be handled by Spring tx.

For adding spring hibernate,

Add dependency of hibernate core.

Orm is the manager to translate the data between spring and hibernate, so we have to include the spring orm dependency.

We can add spring transaction dependency to handle transaction by spring...

Add the following to config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:mvc="http://www.springframework.org/schema/mvc"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
           http://www.springframework.org/schema/mvc
           http://www.springframework.org/schema/tx/spring-mvc.xsd
           http://www.springframework.org/schema/spring-tx
           http://www.springframework.org/schema/tx/spring-tx.xsd
           http://www.springframework.org/schema/context
           https://www.springframework.org/schema/context/spring-context.xsd">

    <context:component-scan base-package="org.example" />

    <!-- Configure view resolver -->
    <bean id="viewResolver"
```

```
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/views/" />
        <property name="suffix" value=".jsp" />
    </bean>

    <mvc:annotation-driven />

    <bean name="dataSource"
class="com.mchange.v2.c3p0.ComboPooledDataSource" destroy-
method="close">
        <property name="driverClass" value="org.postgresql.Driver"/>
        <property name="jdbcUrl" value="jdbc:postgresql://localhost:5432/
prakumar"/>
        <property name="password" value ="prakumar"/>
        <property name="user" value ="prakumar"/>
        <property name="minPoolSize" value="5"/>
        <property name="maxPoolSize" value="10" />
        <property name="maxIdleTime" value="30000"/>
    </bean>

    <bean id="sessionFactory" class
="org.springframework.orm.hibernate5.LocalSessionFactoryBean">
        <property name="dataSource" ref="dataSource" />
<!--        now we need to specify where to search for entities-->
        <property name="packagesToScan" value="org.example.models"/>
        <property name="hibernateProperties">
            <props>
                <prop
key="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</prop>
<!--                i want to see sql queries in the console.-->
                <prop key="show_sql">true</prop>
            </props>
        </property>
    </bean>

    <bean id ="transactionManager"
class="org.springframework.orm.hibernate5.HibernateTransactionManager">
        <property name="sessionFactory" ref="sessionFactory" />
    </bean>

    <tx:annotation-driven transaction-manager="transactionManager"/>

</beans>
```

Hibernate expects that every transaction should be begin and committed else we will get could not get transaction-synchronised session for the current

thread.
We can @Transactional over the method.