DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SUBJECT CODE: 19CS2107
ENTERPRISE PROGRAMMING WORKBOOK

HIBERNATE-SPRING INTEGRATION #12

Date of the Session: ___/___/___ **Time of the Session:** _____to_____

Prerequisite:

- **Basic knowledge on hibernate and spring integration**

Pre-Lab Task:

1. Explain the concept of Integration of Spring-Hibernate Framework.

Answer:- In hibernate framework, we provide all the database information hibernate.cfg.xml file. But if we are going to integrate the hibernate application with spring, we don't need to create the hibernate.cfg.xml file. we can Provide all the information in the applicationContext.xml. file.

2. List out few advantages of Integration.

Answer:- Services layer

The most important benfit is the Spring framework itself. Because of Spring integration, application can leverage all the feautures of spring-framework. For example POJO style Serve interface, IOC, Aspects, Remoting etc.

Hibernate Template :-

Spring provides template for managing sessions, transcation across the application. without the hibernate template, applications need to manage those on their own

Hibernate Dao Support :-

Spring provide class called Hibernate Dao Support. DAO implementation class implements this class to get all the convenience methods that Hibernate Templete provides. If applications are not using Hibernate Templete, then it's good to take advantage of Hibernate Dao Support.

DAO Exception Translation :-

Hibernate 3 throws Runtime exceptions unlike checked exceptions thrown in previous releases.

Spring can translate those exceptions into spring DAO Exceptions and map them into Spring DAO exception hieracty. But you have to use one of the following to get this

1. Use Hibernate Templete
2. @ Repository

Writing space for the Problem:(For Student's use only)

3. List out Methods of Hibernate-template class and write description about it.

Sno	Method	Description
1	void Persist (Objectentity)	Persistes the given object
2	Serializable Save (Objectentity)	Persiste the given Object and returns id
3.	void SaveOr Update (Object entity)	Persists or updates the given object. if id is found it updates the record otherwise Save record
4.	void update (Object entity)	Update the given Object
5.	void delete (Object entity)	Deletes the given Objecton the basis id.
6.	Object get(Class entityClass, Serializable id)	Returns the persistent object on basis of given id.
7.	Object load(Class entity Class, Serilizable id)	Returns persistent Object on the basis of given id.
8.	list load/All(Class entinty Class)	Returns all the Persistent Objects.

In Lab Task:

1. The NavodayaJohar school is wanting to direct a get-together of their 2014-2015 batch students. To design this occasion and oversee, principal appointed a student from a similar batch. To do this the student needed to make hibernate application with spring application. He initially made a table in database to store the subtleties of his companions who are going to the gathering. He made the table with the name Reunion with properties ID No, Name, Contact Number, Amount paid, and Status. ID.NO being the primary key has the most extreme size of 4 digits and Status speaks to how much sum has been paid i.e; completely paid, or partially paid, or not paid. Note that the sum should be paid to go to the gathering is 2000 rupees. Now he needs to make a java project with springs and hibernate integration. Help him in Creating three unique classes for inserting subtleties in database. First make a class for inserting the details of the students who are going to the get-together gathering. Presently make another class for retrieving the amount paid by the students.

PoJo class

```
Package mypackage;
Public class Reunion{
    Private int sid;
    Private String Sname;
    Private doubu contact;
    Private int amount;
    Private String status;

Public int getsid(){
        return sid;}
Public String int void setsid(int sid){
        this.sid = sid;}
Public String getSname(){
        return Sname;}

Public void setSName(){
        this.sname = sname;
}
```

```
Public double getContant(){
        return contact;}

Public void setContact (double contact){
        this.Contact = Contact;
    }
Public int getAmount(){
        return amount;
    }
Public void setAmount (int amount){
        this.amount =amount;
    }
Public String getStatus(){
        return status;}
Public void setStatus (String status){
        this.status = status;
    }
}
```

Mapping file:-

```
<? Xml version="1.0" encoding = "UTF-8"?>
<!Doctype hibernate-mapping PUBLIC
    "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.Sourceforge.net/hibernate-mapping-3.0
dtd">
```

Writing space for the Problem:(For Student's use only)

```
<hibernate-mapping>
<class name = "mypackage.Reunion" table = "reunion">
<id name = "sid">
<generator class = "assigned"> </generator>
</id>
<property name = "Sname"> </property>
<property name = "contact"> </property>
<property name = "amount"> </property>
<property name = "status"> </property>
</class>
</hibernate-mapping>
```

applicationContext.xml :-

```
<?xml version = "1.0" encoding = "UTF-8" ?>

<beans>
<bean id = "datasource" class = "org.apache.commons.
                                 dbcp.BasicDataSource">

<property name = "driverClassName" value = "oracle.jdbc.
                                            driver.OracleDriver">
<property name = "url" value = "jdbc:oracle:thin:@
                                localhost:1552:xe">
                                </property>
<property name = "username" value = "system">
                                </property>
<property name = "password" value = "system">
                                </property>
```

```
</bean>
  <bean id="mysessionfactory" class="org.Springframe
  work.orm.hibernate3.LocalSessionFactoryBean">
  <Property name="SessionFactory" ref="mysessionfactory">
                                    </property>
  </bean>
<bean id="d" class="mypackage.ReunionDao">
    <property name="templete" ref="templete"></property>
  <bean>
    </bean>
```

Insert : javas-
all new

```
Package mypackage;
 import java.util.*;
 import Org.Springframework.beans.factory.BeanFactory;
 import Org.Springframework.beans.factory.xml.XmlBeanFactory;
 import org.Springframework.core.io.ClassPathResource;
 import org.Springframework.core.io.Resource;

Public class Insert {
    Public  static void main(String[] args){
     Resource r= new  ClassPathResource("applicationcontext.xml");
     BeanFactory factory = new XmlBeanFactory(r);
  ReunionDao dao = (ReunionDao) factory.getBean("d");
   Scanner sc = new Scanner(System.in);
   int id;
  String name;
  Double phone;
```

Writing space for the Problem:(For Student's use only)

```
int amt;
String status;
Boolean Continoues =true;
while (Continoues) {
    Reunion e=new Reunion();
System.out.println("Enterth id number:");
id = Sc.neatInt();
System.out.println("Enter the Contact number:");
 Phone = sc.nextDouble();
System.out.println(" Enter the amount tobepaid");
iot =Sc.nextInt();
System.out.println("Enter the status ");
status = Sc.next();
e.setId(id);
e.setName(name);
e.setContact(phone);
e.setAmount(amt);
 e.setStatus(status);
dao.SaveReunion(e);
System.out.println("Do you want to insert(Yes/No)");
String proceed =sc.next();
if(proceed.equals("yes")){
        continous = true;
}
```

```
else {
        continues = false;
    }
  }
}
```

Retrieve.java:-

```
Package mypackage;
import java.util.*;
import org.Springframework.beans.factory.BeanFactory;
import org.Springframework.beans.factory.xml.*;
import org.Springframework.core.io.*;
Public class Retrive{
    Public static void main (String[] args){
        Resource r = new ClassPathResource("applicationContext.xml");
        BeanFactory factory = new XmlBeanFactory(r);

    Reunion e = new Reunion();
System.out.println("Enter the id of the student whose details
are required");
int rid = sc.nextInt();
Reunion h = dao.getById(rid)
System.out.println("the amount paid by student is ");
System.out.println(" Amount Paid "+h.getAmount());
    }
}
```

Post Lab Task:

1. Now to the extension to the last question create a class file in the same java project for updating the database If the student want to pay the amount then update the amount in data base and print the total amount he paid till then and if he had paid the total amount the change the status to fully paid. If not then show the amount that need to be paid by him. Also create a class file for deleting the details of students who cancelled their plan of going go get-together as a result of some issues.

Writing space for the Problem:(For Student's use only)

Update.java

```
Package mypackage;
import java.util.*;
import org.Springframework.beans.factory.*;
import org.Springframework.beans.factory.xml.*;
import org.Springframework.core.io.ClassPathResource;
import org.Springframework.core.io.Resource;

Public class Updated
    Public static void main(String[] args){
Resource r = new ClassPathResource("applicationContext.xml");
BeanFactory factory = new XmlBeanFactory(r);
ReunionDao dao = (ReunionDao) factory.getBean("id");
Scanner sc = new Scanner(System.in);
Boolean continues = true;
```

```
while (continues){

        Reunion e = new Reunion();
System.out.println("Any updates required (Yes/No)");
  String change = sc.next();
if (change.equals("Yes")){
                System.out.println("Enter id number of
    student");
      int uid = sc.nextInt();
  Reunion h = dao.getById(uid);
int cost = h.getAmount();
  System.out.println("the amount paid by the student is :"+cost);
  System.out.println("Do you want to pay (Yes/No)");

  String pay = sc.next();
if (pay.equals("Yes"))
        { System.out.println("Enter the amount
paid now");
  int now = sc.nextInt();
  cost = cost + now;
  h.setAmount(cost);
  dao.update Reunion(h);
    if (cost == 1000) {
            h.setStatus("fully paid");
              dao.updateReunion(h);
```

Writing space for the Problem:(For Student's use only)

```
}
}
else { System.out.println("the amount Student has to
                    Pay" + (2000 + ) } }

  else
      {
        System.out.println("Update are done") ;
        Continues = false; } } }

Delete.java
ARUN.JUV

Package myPackage;
import java.util.*;
import Org.Springframe.beans.factory.*;
import org.Springframework.core.io.*;

Public class Delete {
        Public static void main (String() args)
        {
        Resource r = new ClassPathResource
                        ("application.xml");

        while (Continues){
```

```
System.out.println("Enter the id of student who
wants to drop from attending gathering");
        int did=Sc.nextInt();
        Reunion w = dao.getById(did);
          dao.deleteReunion(w);
        System.out.println("the details of student
                          are deleted");
System.out.println("Do you want to delete some
          more  yes/no)");
    String del = Sc.next();
        if (del.equals("yes")) {
                    continues=true;
                }
            else { continues=false;}
```