**Jsf crud**

**Jar file:**



**Hibernate cfg.xml Page**

<session-factory>

<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

<property name="hibernate.connection.driver\_class">com.mysql.jdbc.Driver</property>

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/jsfcrud?zeroDateTimeBehavior=convertToNull</property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">1234</property>

<property name="hibernate.hbm2ddl.auto">update</property>

<property name="hibernate.current\_session\_context\_class">thread</property>

<property name="show\_sql">true</property>

<mapping class="entity.Student"/>

</session-factory>

**Student**

package entity;

import java.util.Date;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table

public class Student {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

@Column(name = "name", nullable = false)

private String name;

private String gender;

private String round;

private Date startingDate;

private boolean status;

private String[] completedCourse;

private String notes;

public Student() {

}

public Student(int id, String name, String gender, String round, Date startingDate, boolean status, String[] completedCourse, String notes) {

this.id = id;

this.name = name;

this.gender = gender;

this.round = round;

this.startingDate = startingDate;

this.status = status;

this.completedCourse = completedCourse;

this.notes = notes;

}

public Student(String name, String gender, String round, Date startingDate, boolean status, String[] completedCourse, String notes) {

this.name = name;

this.gender = gender;

this.round = round;

this.startingDate = startingDate;

this.status = status;

this.completedCourse = completedCourse;

this.notes = notes;

}

public int getId() {

return id;

}

public String getName() {

return name;

}

public String getGender() {

return gender;

}

public String getRound() {

return round;

}

public Date getStartingDate() {

return startingDate;

}

public boolean isStatus() {

return status;

}

public String[] getCompletedCourse() {

return completedCourse;

}

public String getNotes() {

return notes;

}

public void setId(int id) {

this.id = id;

}

public void setName(String name) {

this.name = name;

}

public void setGender(String gender) {

this.gender = gender;

}

public void setRound(String round) {

this.round = round;

}

public void setStartingDate(Date startingDate) {

this.startingDate = startingDate;

}

public void setStatus(boolean status) {

this.status = status;

}

public void setCompletedCourse(String[] completedCourse) {

this.completedCourse = completedCourse;

}

public void setNotes(String notes) {

this.notes = notes;

}

@Override

public String toString() {

return "Student{" + "id=" + id + ", name=" + name + ", gender=" + gender + ", round=" + round + ", startingDate=" + startingDate + ", status=" + status + ", completedCourse=" + completedCourse + ", notes=" + notes + '}';

}

}

**StudentService Page**

package service;

import entity.Student;

import java.util.List;

public interface StudentService {

void save(Student student);

void update(Student student);

void delete(Student student);

List<Student> getStudents();

}

**StudentServiceImpl Page**

package serviceImpl;

import entity.Student;

import java.util.ArrayList;

import java.util.List;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import service.StudentService;

import util.NewHibernateUtil;

public class StudentServiceImpl implements StudentService {

@Override

public void save(Student student) {

SessionFactory sf = NewHibernateUtil.getSessionFactory();

Session sn = sf.getCurrentSession();

Transaction tr = sn.beginTransaction();

sn.save(student);

tr.commit();

}

@Override

public void update(Student student) {

SessionFactory sf = NewHibernateUtil.getSessionFactory();

Session sn = sf.getCurrentSession();

Transaction tr = sn.beginTransaction();

sn.saveOrUpdate(student);

tr.commit();

}

@Override

public void delete(Student student) {

SessionFactory sf = NewHibernateUtil.getSessionFactory();

Session sn = sf.getCurrentSession();

Transaction tr = sn.beginTransaction();

sn.delete(student);

tr.commit();

}

@Override

public List<Student> getStudents() {

List<Student> list = new ArrayList<>();

SessionFactory sf = NewHibernateUtil.getSessionFactory();

Session sn = sf.getCurrentSession();

Transaction tr = sn.beginTransaction();

list = sn.createCriteria(Student.class).list();

tr.commit();

return list;

}

}

**StudentController Page**

package controller;

import entity.Student;

import java.util.ArrayList;

import java.util.List;

import javax.faces.application.FacesMessage;

import javax.faces.bean.ManagedBean;

import javax.faces.context.FacesContext;

import service.StudentService;

import serviceImpl.StudentServiceImpl;

@ManagedBean

public class StudentController {

private Student student;

private List<Student> students;

private StudentService service;

public void save() {

service = new StudentServiceImpl();

service.save(student);

FacesContext.getCurrentInstance().addMessage("MessageId", new FacesMessage(FacesMessage.SEVERITY\_INFO, "Save Successfull!", null));

System.out.println("Saved Successfully !");

}

public void update() {

service = new StudentServiceImpl();

service.update(student);

FacesContext.getCurrentInstance().addMessage("MessageIdUp", new FacesMessage(FacesMessage.SEVERITY\_INFO, "Update Successfull!", null));

System.out.println("Updated Successfully !");

}

public void delete() {

service = new StudentServiceImpl();

service.delete(student);

FacesContext.getCurrentInstance().addMessage("MessageIdUp", new FacesMessage(FacesMessage.SEVERITY\_INFO, "Delete Successfull!", null));

System.out.println("Deleted Successfully !");

}

public Student getStudent() {

if (student == null) {

student = new Student();

}

return student;

}

public void setStudent(Student student) {

this.student = student;

}

public List<Student> getStudentsList() {

students = new ArrayList<>();

service = new StudentServiceImpl();

students = service.getStudents();

return students;

}

public void setStudentsList(List<Student> studentsList) {

this.students = studentsList;

}

}

**Index.xhtml Page**

<h:body>

<div style="width: 90%; margin: auto; text-align: center; font-size: xx-large; color: green">

Welcome to Student Data Store

</div>

<div style="width: 90%; margin: auto">

<h:form id="sh">

<p:messages for="MessageId" showDetail="true" autoUpdate="true" closable="true"/>

<p:messages for="MessageIdUp" showDetail="false" autoUpdate="true" closable="true"/>

<p:outputLabel value="ID: "/>

<p:inputText value="#{studentController.student.id}"/><br/>

<p:outputLabel value="Name: "/>

<p:inputText value="#{studentController.student.name}"/><br/>

<p:outputLabel value="Gender: "/>

<p:selectOneRadio id="gender" value="#{studentController.student.gender}">

<f:selectItem itemLabel="Male" itemValue="Male" />

<f:selectItem itemLabel="Female" itemValue="Female" />

</p:selectOneRadio><br/>

<p:outputLabel value="Round: "/>

<p:selectOneMenu id="round" value="#{studentController.student.round}" style="width:125px">

<f:selectItem itemLabel="Select One" itemValue="" />

<f:selectItem itemLabel="Round 37" itemValue="Round 37" />

<f:selectItem itemLabel="Round 38" itemValue="Round 38" />

<f:selectItem itemLabel="Round 39" itemValue="Round 39" />

</p:selectOneMenu><br/>

<p:outputLabel value="Completed Course: "/>

<p:selectManyCheckbox id="course" value="#{studentController.student.completedCourse}">

<f:selectItem itemLabel="HTML" itemValue="HTML" />

<f:selectItem itemLabel="Oracle" itemValue="Oracle" />

<f:selectItem itemLabel="Java" itemValue="Java" />

</p:selectManyCheckbox><br/>

<p:outputLabel value="Status: "/>

<p:selectOneMenu id="status" value="#{studentController.student.status}" style="width:125px">

<f:selectItem itemLabel="Running" itemValue="Running" />

<f:selectItem itemLabel="Completed" itemValue="Completed" />

</p:selectOneMenu><br/>

<p:outputLabel value="Date: "/>

<p:calendar value="#{studentController.student.startingDate}"/><br/>

<p:outputLabel value="Notes: "/>

<p:inputText value="#{studentController.student.notes}"/><br/>

<p:commandButton action="#{studentController.save()}" value="Save" update=":sh :lists:stuList"/>

<p:commandButton action="#{studentController.update()}" value="Update" update=":sh :lists:stuList"/>

<p:commandButton action="#{studentController.delete()}" value="Delete" update=":lists:stuList"/>

<h:commandButton value="Reset" style="margin-right:25px;" >

<p:ajax update="sh" resetValues="true" />

</h:commandButton>

</h:form>

</div><hr/>

<div style="width: 90%; margin: auto">

<h:form id="lists">

<p:dataTable id="stuList" var="stu" value="#{studentController.studentsList}">

<p:column headerText="ID">

<h:outputText value="#{stu.id}" />

</p:column>

<p:column headerText="Name">

<h:outputText value="#{stu.name}" />

</p:column>

<p:column headerText="Gender">

<h:outputText value="#{stu.gender}" />

</p:column>

<p:column headerText="Round">

<h:outputText value="#{stu.round}" />

</p:column>

<p:column headerText="Completed Course">

<p:dataList var="course" value="#{stu.completedCourse}" type="ordered">

<h:outputText value="#{course}"/>

</p:dataList>

</p:column>

<p:column headerText="Status">

<h:outputText value="#{stu.status}" />

</p:column>

<p:column headerText="Starting Date">

<h:outputText value="#{stu.startingDate}">

<f:convertDateTime pattern="dd.MM.yyyy" />

</h:outputText>

</p:column>

<p:column headerText="Notes">

<h:outputText value="#{stu.notes}" />

</p:column>

</p:dataTable>

</h:form>

</div>

</h:body>