**Practical no-7**

**Student.java**

import java.io.Serializable;

import javax.persistence.Basic;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.NamedQueries;

import javax.persistence.NamedQuery;

import javax.persistence.Table;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Size;

import javax.xml.bind.annotation.XmlRootElement;

@Entity

@Table(name = "student")

@XmlRootElement

@NamedQueries({

@NamedQuery(name = "Student.findAll", query = "SELECT s FROM Student s")

, @NamedQuery(name = "Student.findByStudId", query = "SELECT s FROM Student s WHERE s.studId = :studId")

, @NamedQuery(name = "Student.findByStudName", query = "SELECT s FROM Student s WHERE s.studName = :studName")

, @NamedQuery(name = "Student.findByStudMarks", query = "SELECT s FROM Student s WHERE s.studMarks = :studMarks")})

public class Student implements Serializable {

private static final long serialVersionUID = 1L;

@Id

@Basic(optional = false)

@NotNull

@Column(name = "stud\_id")

private Integer studId;

@Size(max = 15)

@Column(name = "stud\_name")

private String studName;

@Column(name = "stud\_marks")

private Integer studMarks;

public Student() {

}

public Student(Integer studId) {

this.studId = studId;

}

public Integer getStudId() {

return studId;

}

public void setStudId(Integer studId) {

this.studId = studId;

}

**Roll no-59**

public String getStudName() {

return studName;

}

public void setStudName(String studName) {

this.studName = studName;

}

public Integer getStudMarks() {

return studMarks;

}

public void setStudMarks(Integer studMarks) {

this.studMarks = studMarks;

}

@Override

public int hashCode() {

int hash = 0;

hash += (studId != null ? studId.hashCode() : 0);

return hash;

}

@Override

public boolean equals(Object object) {

// TODO: Warning - this method won't work in the case the id fields are not set

if (!(object instanceof Student)) {

return false;

}

Student other = (Student) object;

if ((this.studId == null && other.studId != null) || (this.studId != null && !this.studId.equals(other.studId))) {

return false;

}

return true;

}

@Override

public String toString() {

return "demoPackage.Student[ studId=" + studId + " ]";

}

}

**StudentResource.java**

import javax.persistence.EntityManager;

import javax.persistence.EntityManagerFactory;

import javax.persistence.Persistence;

import javax.persistence.PersistenceContext;

import javax.persistence.criteria.CriteriaQuery;

import javax.ws.rs.core.Context;

import javax.ws.rs.core.UriInfo;

import javax.ws.rs.Consumes;

import javax.ws.rs.DELETE;

import javax.ws.rs.Produces;

import javax.ws.rs.GET;

import javax.ws.rs.POST;

import javax.ws.rs.PUT;

import javax.ws.rs.Path;

import javax.ws.rs.PathParam;

import javax.ws.rs.core.MediaType;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

@Path("student")

public class StudentResource {

@PersistenceContext(unitName="StudentAPIPU")

private final EntityManager em;

private final EntityManagerFactory emf;

Connection conn;

Statement stmt;

/\*\*

\* Creates a new instance of StudentResource

\*/

public StudentResource() {

emf=Persistence.createEntityManagerFactory("StudentAPIPU");

em=emf.createEntityManager();

try{

Class.forName("com.mysql.jdbc.Driver");

conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/studentapi","root","");

stmt=conn.createStatement();

}catch(Exception e){

//check the error message in glassfish tab

System.out.println(e.getMessage());

}

}

@GET

@Produces(MediaType.APPLICATION\_JSON)

public String getStudentDetails() throws JSONException {

List<Student> students= em.createNamedQuery("Student.findAll").getResultList();

JSONObject mainObj=new JSONObject();

JSONArray studentJsonArray=new JSONArray();

for(Student student:students){

JSONObject studObj=new JSONObject();

studObj.put("stud\_id",student.getStudId());

studObj.put("stud\_name",student.getStudName());

studObj.put("stud\_marks",student.getStudMarks());

studentJsonArray.put(studObj);

}

mainObj.put("Student",studentJsonArray);

return mainObj.toString();

}

@POST

@Path("/putStudentDetails/ {stud\_id},{stud\_name},{stud\_marks}")

@Consumes(MediaType.APPLICATION\_JSON)

public void putStudentDetails(@PathParam("stud\_id") String stud\_id,@PathParam("stud\_name") String stud\_name,@PathParam("stud\_marks") int stud\_marks) {

try{

stmt.executeUpdate("insert into student(stud\_id,stud\_name,stud\_marks) values("+stud\_id+",'"+stud\_name+"',"+stud\_marks+");");

}

catch(Exception e){

System.out.println(e.getMessage());

}

}

@Path("/deleteStudent/{stud\_id}")

//@DELETE //notwroking with delete

@POST

@Consumes(MediaType.APPLICATION\_JSON)

public void deleteStudent(@PathParam("stud\_id") String stud\_id){

try{

stmt.executeUpdate("delete from student where stud\_id="+stud\_id);

}

catch(Exception e){

System.out.println(e.getMessage());

}

}

@PUT

@Path("/updateStudentNameById/{stud\_id},{stud\_name}")

@Consumes(MediaType.APPLICATION\_JSON)

public void updateStudentNameById(@PathParam("stud\_id") String stud\_id,@PathParam("stud\_name") String new\_stud\_name){

try{

stmt.executeUpdate("update student set stud\_name='"+new\_stud\_name+"' where stud\_id="+stud\_id);

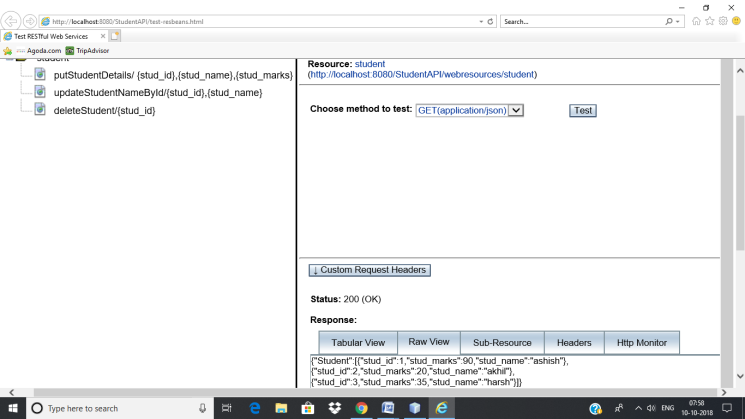
}

catch(Exception e){}}

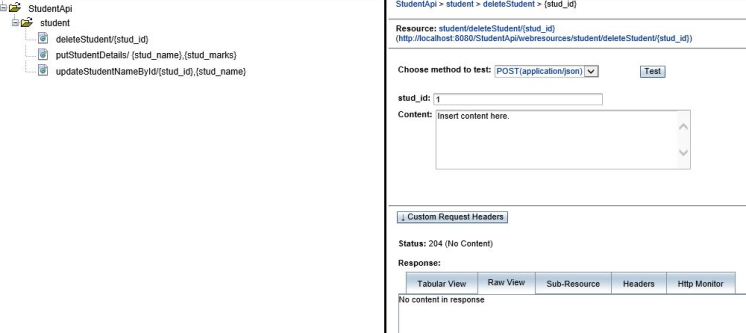
System.out.println(e.getMessage());

}

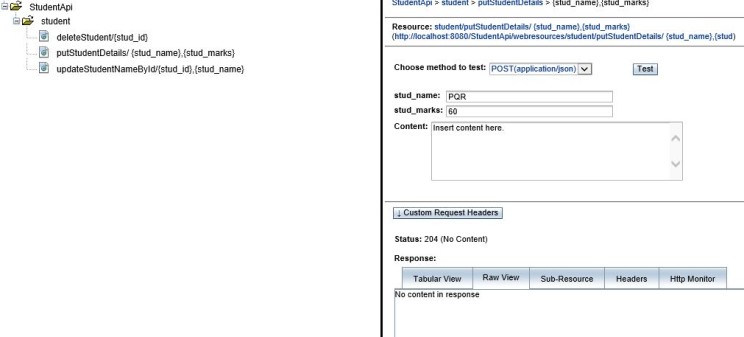
**OUTPUT:**



Delete student:



Insert student:



Update Student:

