**Experiment 9**

**Student Name: Charanjeet Singh UID:22BCS16411**

**Branch: BE-CSE Section/Group:22BCSIOT-616-B**

**Semester:6th Date of Performance:21/04/25**

**Subject Name: Project Based Learning Subject Code: 22CSH-359 in Java with Lab**

1.EASY LEVEL:

public class Course {

private String courseName;

private int duration;

public Course(String courseName, int duration) {

this.courseName = courseName;

this.duration = duration;

}

public String getDetails() {

return courseName + " - " + duration + " months";

}

}

public class Student {

private String name;

private Course course;

public Student(String name, Course course) {

this.name = name;

this.course = course;

}

public void displayInfo() {

System.out.println("Name: " + name);

System.out.println("Course: " + course.getDetails());

}

}

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class AppConfig {

@Bean

public Course course() {

return new Course("Java Spring", 3);

}

@Bean

public Student student() {

return new Student("Ravi Kumar", course());

}

}

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class MainApp {

public static void main(String[] args) {

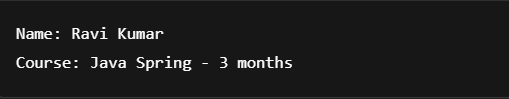
ApplicationContext context = new AnnotationConfigApplicationContext(AppConfig.class);

Student student = context.getBean(Student.class);

student.displayInfo();

}

}



2. Medium level:

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

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

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/your\_db</property>

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

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

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

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

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

<mapping class="Student"/>

</session-factory>

</hibernate-configuration>

import jakarta.persistence.\*;

@Entity

public class Student {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

private int age;

public Student() {}

public Student(String name, int age) {

this.name = name;

this.age = age;

}

// Getters and Setters

}

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

public class StudentDAO {

private static SessionFactory factory = new Configuration().configure().buildSessionFactory();

public void createStudent(Student s) {

Session session = factory.openSession();

session.beginTransaction();

session.save(s);

session.getTransaction().commit();

session.close();

}

public void updateStudent(int id, String name) {

Session session = factory.openSession();

session.beginTransaction();

Student s = session.get(Student.class, id);

s.setName(name);

session.update(s);

session.getTransaction().commit();

session.close();

}

public void deleteStudent(int id) {

Session session = factory.openSession();

session.beginTransaction();

Student s = session.get(Student.class, id);

session.delete(s);

session.getTransaction().commit();

session.close();

}

public Student readStudent(int id) {

Session session = factory.openSession();

return session.get(Student.class, id);

}

}

public class MainApp {

public static void main(String[] args) {

StudentDAO dao = new StudentDAO();

Student s1 = new Student("Ankit", 22);

dao.createStudent(s1);

dao.updateStudent(1, "Ankit Verma");

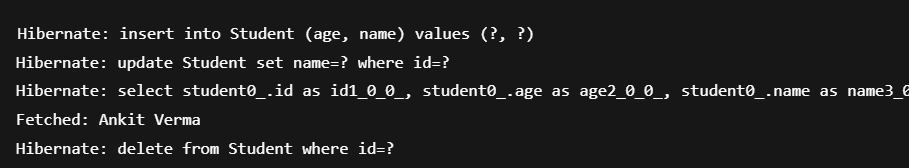
Student student = dao.readStudent(1);

System.out.println("Fetched: " + student.getName());

dao.deleteStudent(1);

}

}



3.Hard Level:

import jakarta.persistence.\*;

@Entity

public class Account {

@Id

private int id;

private String name;

private double balance;

// Constructors, Getters, Setters

}

import jakarta.persistence.\*;

import java.util.Date;

@Entity

public class Transaction {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private int fromAccount;

private int toAccount;

private double amount;

private Date timestamp = new Date();

// Constructors, Getters, Setters

}

import org.springframework.context.annotation.\*;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

@Configuration

public class AppConfig {

@Bean

public SessionFactory sessionFactory() {

return new Configuration().configure().buildSessionFactory();

}

@Bean

public BankService bankService() {

return new BankService(sessionFactory());

}

}

import org.hibernate.\*;

public class BankService {

private SessionFactory factory;

public BankService(SessionFactory factory) {

this.factory = factory;

}

public void transfer(int fromId, int toId, double amount) {

Session session = factory.openSession();

Transaction tx = null;

try {

tx = session.beginTransaction();

Account from = session.get(Account.class, fromId);

Account to = session.get(Account.class, toId);

if (from.getBalance() < amount) {

throw new RuntimeException("Insufficient balance");

}

from.setBalance(from.getBalance() - amount);

to.setBalance(to.getBalance() + amount);

session.update(from);

session.update(to);

Transaction t = new Transaction(fromId, toId, amount);

session.save(t);

tx.commit();

System.out.println("Transfer successful.");

} catch (Exception e) {

if (tx != null) tx.rollback();

System.out.println("Transfer failed: " + e.getMessage());

} finally {

session.close();

}

}

}

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class MainApp {

public static void main(String[] args) {

ApplicationContext context = new AnnotationConfigApplicationContext(AppConfig.class);

BankService service = context.getBean(BankService.class);

service.transfer(1, 2, 1000);

}

}

