Experiment -9

Student Name: Aditya Mehta **UID:22BCS17094**
Branch: BE-CSE **Section/Group:IOT-642-B**
Semester:6th **Date of Performance:7/04/2025**
Subject Name: Project-Based Learning in Java with Lab **Subject Code: 22CSH-359**

9.1.1.Aim: To demonstrate dependency injection using Spring Framework with Java-based configuration.

9.1.2 Objective:

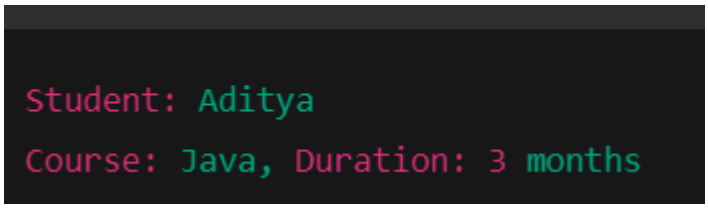
Define Course and Student classes.
Use Configuration and Bean annotations to inject dependencies. Load Spring context and print student details.

9.1.3 Code:

```
// Course.java public class Course { private String courseName; private String duration;

   public Course(String courseName, String duration) { this.courseName = courseName; this.duration = duration;
   }

   public String getCourseName() { return courseName; } public String getDuration() { return duration; }

   @Override
   public String toString() {
      return "Course: " + courseName + ", Duration: " + duration;
   }
}

// Student.java public class Student { private
```

```
String name;     private
Course course;     public
Student(String name,
Course course) {
        this.name = name;
this.course = course;
    }

    public void showDetails() {
        System.out.println("Student: " + name);
        System.out.println(course);
    }
}// AppConfig.java
import org.springframework.context.annotation.*;

@Configuration public
class AppConfig {
    @Bean
    public Course course() {
        return new Course("Java", "3 months");
    }

    @Bean
    public Student student() {
        return new Student("Aditya", course());
    }
}// MainApp.java
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.showDetails();
    } }
```

Output:



9.2.1 Aim: To perform CRUD operations on a Student entity using Hibernate ORM with MySQL.

Objective: Define Course and Student classes.

Use Configuration and Bean annotations to inject dependencies.

Load Spring context and print student details.

9.2.2 Code:

```
<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/testdb</property>
    <property name="hibernate.connection.username">root</property>
    <property name="hibernate.connection.password">password</property>
    <property
name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
    <property name="hibernate.hbm2ddl.auto">update</property>
    <mapping class="Student"/>
  </session-factory>
</hibernate-configuration>

import javax.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, setters, toString
} import
org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class HibernateUtil {
    private static final SessionFactory sessionFactory;
     static
{
        sessionFactory = new Configuration().configure().buildSessionFactory();
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}

import org.hibernate.*;

public class MainCRUD {
    public static void main(String[] args) {
        Session session = HibernateUtil.getSessionFactory().openSession();

        // Create
        Transaction tx = session.beginTransaction();
Student s1 = new Student("Aman", 22);
        session.save(s1);
        tx.commit();
```

```
        // Read
        Student student = session.get(Student.class, 1);
        System.out.println(student);

        // Update
        tx = session.beginTransaction();
student.setAge(23);
session.update(student);        tx.commit();

        // Delete
        tx = session.beginTransaction();
session.delete(student);
    tx.commit();

        session.close();
    }
 }
```

9.2.3 Output:

```
Student s1 = new Student("Aditya", 22);
s1.setId(17094);
```