Hibernate Many-to-Many Example (Student ↔ Course)

This project demonstrates **Hibernate Many-to-Many Mapping** using annotations with `Student` and `Course` entities.

🚀 Steps to Create Maven Project

1. **Open IDE (IntelliJ / Eclipse / VS Code with Maven plugin).**
2. **Create New Maven Project.**
   - In Eclipse: *File → New → Maven Project*
   - In IntelliJ: *File → New → Project → Maven*
3. Provide **GroupId** (e.g., `com.example`) and **ArtifactId** (e.g., `hibernate-many-to-many`).
4. Add the `pom.xml` configuration given below.
5. Create package structure:
   - `src/main/java/com/example` → For Java classes
   - `src/main/resources` → For `hibernate.cfg.xml`
6. Add **Entity Classes** (`Student.java`, `Course.java`).
7. Add **HibernateUtil.java** for session management.
8. Add a **MainApp.java** to test CRUD operations.
9. Run the project. Hibernate will create/update database tables automatically.

📌 pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
         http://maven.apache.org/xsd/maven-4.0.0.xsd">

    <modelVersion>4.0.0</modelVersion>
    <groupId>com.example</groupId>
    <artifactId>hibernate-many-to-many</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>17</maven.compiler.source>
        <maven.compiler.target>17</maven.compiler.target>
        <hibernate.version>6.2.7.Final</hibernate.version>
    </properties>

    <dependencies>
        <!-- Hibernate Core -->
        <dependency>
```

```
            <groupId>org.hibernate.orm</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>${hibernate.version}</version>
        </dependency>

        <!-- MySQL Connector -->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-j</artifactId>
            <version>8.3.0</version>
        </dependency>

        <!-- Jakarta Persistence API (JPA) -->
        <dependency>
            <groupId>jakarta.persistence</groupId>
            <artifactId>jakarta.persistence-api</artifactId>
            <version>3.1.0</version>
        </dependency>

        <!-- JUnit for Testing -->
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.13.2</version>
            <scope>test</scope>
        </dependency>

        <!-- Logging -->
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-api</artifactId>
            <version>2.0.12</version>
        </dependency>
        <dependency>
            <groupId>ch.qos.logback</groupId>
            <artifactId>logback-classic</artifactId>
            <version>1.4.14</version>
        </dependency>
    </dependencies>
</project>
```

📌 hibernate.cfg.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
```

```
        <!-- Database Connection -->
        <property
name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</propert
y>
        <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/hibernate_demo
</property>
        <property name="hibernate.connection.username">root</property>
        <property
name="hibernate.connection.password">yourpassword</property>

        <!-- Hibernate Properties -->
        <property
name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property name="hibernate.hbm2ddl.auto">update</property>
        <property name="hibernate.show_sql">true</property>
        <property name="hibernate.format_sql">true</property>

        <!-- Entity Classes -->
        <mapping class="com.example.Student"/>
        <mapping class="com.example.Course"/>
    </session-factory>
</hibernate-configuration>
```

📌 Student.java

```
import jakarta.persistence.*;
import java.util.HashSet;
import java.util.Set;

@Entity
@Table(name = "students")
public class Student {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column(nullable = false)
    private String name;

    @Column(unique = true)
    private String email;

    @ManyToMany(cascade = {CascadeType.ALL})
    @JoinTable(
        name = "student_courses",
        joinColumns = { @JoinColumn(name = "student_id") },
        inverseJoinColumns = { @JoinColumn(name = "course_id") }
    )
```

```
    private Set<Course> courses = new HashSet<>();

    public Student() {}
    public Student(String name, String email) {
        this.name = name;
        this.email = email;
    }

    public void enrollCourse(Course course) {
        courses.add(course);
        course.getStudents().add(this);
    }

    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public String getEmail() { return email; }
    public void setEmail(String email) { this.email = email; }

    public Set<Course> getCourses() { return courses; }
    public void setCourses(Set<Course> courses) { this.courses = courses;
}
}
```

📌 Course.java

```
import jakarta.persistence.*;
import java.util.HashSet;
import java.util.Set;

@Entity
@Table(name = "courses")
public class Course {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column(nullable = false, unique = true)
    private String title;

    private int credits;

    @ManyToMany(mappedBy = "courses")
    private Set<Student> students = new HashSet<>();
```

```
    public Course() {}
    public Course(String title, int credits) {
        this.title = title;
        this.credits = credits;
    }

    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getTitle() { return title; }
    public void setTitle(String title) { this.title = title; }

    public int getCredits() { return credits; }
    public void setCredits(int credits) { this.credits = credits; }

    public Set<Student> getStudents() { return students; }
    public void setStudents(Set<Student> students) { this.students =
students; }
}
```

📌 HibernateUtil.java

```
package com.example.util;

import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
import org.hibernate.service.ServiceRegistry;

import com.example.Student;
import com.example.Course;

public class HibernateUtil {

    private static SessionFactory sessionFactory;

    static {
        try {
            Configuration configuration = new Configuration();
            configuration.configure("hibernate.cfg.xml");

            configuration.addAnnotatedClass(Student.class);
            configuration.addAnnotatedClass(Course.class);

            ServiceRegistry serviceRegistry = new
StandardServiceRegistryBuilder()
                    .applySettings(configuration.getProperties())
                    .build();
```

```
            sessionFactory =
configuration.buildSessionFactory(serviceRegistry);

        } catch (Throwable ex) {
            System.err.println("SessionFactory creation failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }

    public static void shutdown() {
        getSessionFactory().close();
    }
}
```

📌 MainApp.java

```
import org.hibernate.Session;
import org.hibernate.Transaction;

import com.example.Student;
import com.example.Course;
import com.example.util.HibernateUtil;

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

        Course course1 = new Course("Hibernate ORM", 4);
        Course course2 = new Course("Spring Boot", 3);
        Course course3 = new Course("Data Structures", 5);

        Student student1 = new Student("Alice", "alice@example.com");
        Student student2 = new Student("Bob", "bob@example.com");

        student1.enrollCourse(course1);
        student1.enrollCourse(course2);

        student2.enrollCourse(course2);
        student2.enrollCourse(course3);

        session.save(student1);
        session.save(student2);

        tx.commit();
        session.close();
```

```
            HibernateUtil.shutdown();
        }
    }
```