Implementasi Method selectAllStudents()

Perubahan pada method tersebut dilakukan untuk dapat menampilkan daftar mata kuliah yang diikuti masing-masing student. Untuk itu dilakukan perubahan Model seperti pada contoh tutorial, dan perubahan pada Mapper serta View.

```
@Select("SELECT npm, name, gpa FROM student")
@Results(value = {
        @Result(property="npm", column="npm"),
        @Result(property="name", column="name"),
        @Result(property="gpa", column="gpa"),
        @Result(property="courses", column="npm",
        javaType = List.class,
        many=@Many(select="selectStudentCourses"))
        })
List<StudentModel> selectAllStudents ();
```

Pada Mapper untuk melakukan selectAllStudents(), diambil list dari student yang berisi data npm, nama, dan gpa, serta list dari courses yang diambil student tersebut. Untuk mendapatkan list courses tersebut, dilakukan chaining ke method selectStudentCourses().

```
@Select("SELECT course.id_course, name, credits " +
        "FROM studentcourse JOIN course " +
        "ON studentcourse.id_course = course.id_course " +
        "WHERE studentcourse.npm = #{npm}")
List<CourseModel> selectStudentCourses (@Param("npm") String npm);
```

Method tersebut mengembalikan list courses yang diambil seorang students dengan melakukan operasi join tabel studentcourses dan course. Tabel studentcourses untuk memberikan parameter npm student, dan tabel course untuk mendapatkan data id_course, nama, serta kredit course tersebut.

Selain itu diperlukan juga penyesuaian pada view dengan melakukan iterasi list courses pada masing-masing student.

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <title>View All Students</title>
    </head>
    <body>
        <h1>All Students</h1>

        <div th:each="student,iterationStatus: ${students}">
            <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>
            <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
            <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
            <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
            <ul th:each="course,iterationStatus: ${student.courses}">
                <li th:text="${course.name} + '-' + ${course.credits} + ' sks'" >
                    Nama kuliah-X SKS
                </li>
            </ul>
            <a th:href="'/student/delete/' + ${student.npm}" > Delete Data</a><br/>
            <a th:href="'/student/update/' + ${student.npm}" > Update Data</a><br/>
            <hr/>
        </div>
    </body>
</html>
```

Implementasi Method viewCourse()

Untuk dapat melakukan implementasi viewCourse(), terdapat berbagai code yang harus ditambahkan, baik itu pada Controller, Service, Mapper, maupun View. Perubahan pada Mapper dilakukan dengan menambahkan method selectCourseParticipant() dan selectCourse().

```
@Select("SELECT id_course, name, credits FROM course WHERE id_course = #{id}")
@Results(value = {
        @Result(property="id_course", column="id_course"),
        @Result(property="name", column="name"),
        @Result(property="credits", column="credits"),
        @Result(property="students", column="id_course",
        javaType = List.class,
        many=@Many(select="selectCourseParticipant"))
        })
CourseModel selectCourse(@Param("id") String id);

@Select("SELECT student.npm, student.name, student.gpa " +
        "FROM studentcourse JOIN student " +
        "ON studentcourse.npm = student.npm " +
        "WHERE studentcourse.id_course = #{id}")
List<StudentModel> selectCourseParticipant (@Param("id") String id);
```

Method selectCourse() digunakan untuk mendapatkan data sebuah course berupa id_course, nama, dan jumlah kredit course tersebut beserta dengan list students yang mengambil coursenya. Untuk mendapatkan list students tersebut dilakukan chaining ke method selectCourseParticipant() yang melakukan join tabel students dan studentcourse untuk mendapatkan list students yang mengambil suatu course.

Terdapat pula tambahan code pada Service untuk menghubungkan hasil mapping ke Controller. Pada interface ditambahkan method sebagai berikut:

```
public interface StudentService
{
    StudentModel selectStudent (String npm);

    List<StudentModel> selectAllStudents ();

    void addStudent (StudentModel student);

    void deleteStudent (String npm);

    void updateStudent (StudentModel student);

    CourseModel selectCourse (String id);
}
```

Sedangkan pada implementasinya dilakukan logging dan pengiriman data hasil mapping ke Controller.

```
@Override
public CourseModel selectCourse(String id_course) {
    log.info ("select course with id_course {}", id_course);
    return studentMapper.selectCourse(id_course);
}
```

Selain itu, penambahan code pada Controller dilakukan dengan menambahkan @RequestMapping untuk route ”/course/view/{id}”.

```java
@RequestMapping("/course/view/{id_course}")
public String viewCourse(Model model, @PathVariable(value="id_course") String id_course) {
    CourseModel course = studentDAO.selectCourse(id_course);

    if (course != null) {
        model.addAttribute ("course", course);
        return "viewcourse";
    } else {
        return "course-not-found";
    }
}
```

Mapping tersebut menerima @PathVariable berupa id_course, dan melakukan validasi apakah course tersebut ada atau tidak dengan menjalankan service selectCourse() yang sudah dibuat sebelumnya. Jika ada maka akan ditampilkan detail course tersebut beserta dengan list participantnya. Jika tidak maka akan ditampilkan halaman pemberitahuan bahwa course not found. Adapun View untuk menampilkan detail course adalah sebagai berikut:

```html
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <title>View Course Participant</title>
    </head>
    <body>
        <div>
            <h3 th:text="'ID = ' + ${course.id_course}">Course ID</h3>
            <h3 th:text="'Nama = ' + ${course.name}">Course Name</h3>
            <h3 th:text="'SKS = ' + ${course.credits}">Course Credits</h3>
            <h3>Mahasiswa yang mengambil</h3>
            <ul th:each="students,iterationStatus: ${course.students}">
                <li th:text="${students.npm} + ' - ' + ${students.name}" >
                    Nama kuliah-X SKS
                </li>
            </ul>
        </div>
    </body>
</html>
```

View menampilkan data course dan melakukan looping pada list students yang mengambil course tersebut dan menampilkan npm serta nama student. Sedangkan pada View untuk course not found hanya menampilkan seperti student not found.

```html
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <title>Course not found</title>
    </head>
    <body>
        <h1>Course not found</h1>
        <h3 th:text="'Course ID = ' + ${id_course}">Course ID</h3>
    </body>
</html>
```