

Pada tutorial kali ini, yang dipelajari adalah bagaimana membuat web dengan SpringBoot dan berhubungan langsung dengan database. Adanya annotation baru diketahui (@Result) bisa membantu untuk memetakan hasil dari query ke model dan mengetahui annotation @Many yang bisa mengambil data pada course dan menghubungkannya dengan student. Dan mengetahui penggunaan MyBatis dan Lombok.

Latihan

1. Ubah method selectAllStudents pada kelas StudentMapper agar halaman viewall menampilkan semua student beserta daftar kuliah yang diambil.

```
@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 = "selectCourses")) })
List<StudentModel> selectAllStudents ();
```

Pada method ini, dilakukan penambahan annotation baru yaitu Result, yang menghubungkan student course dengan student berdasarkan npm.

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

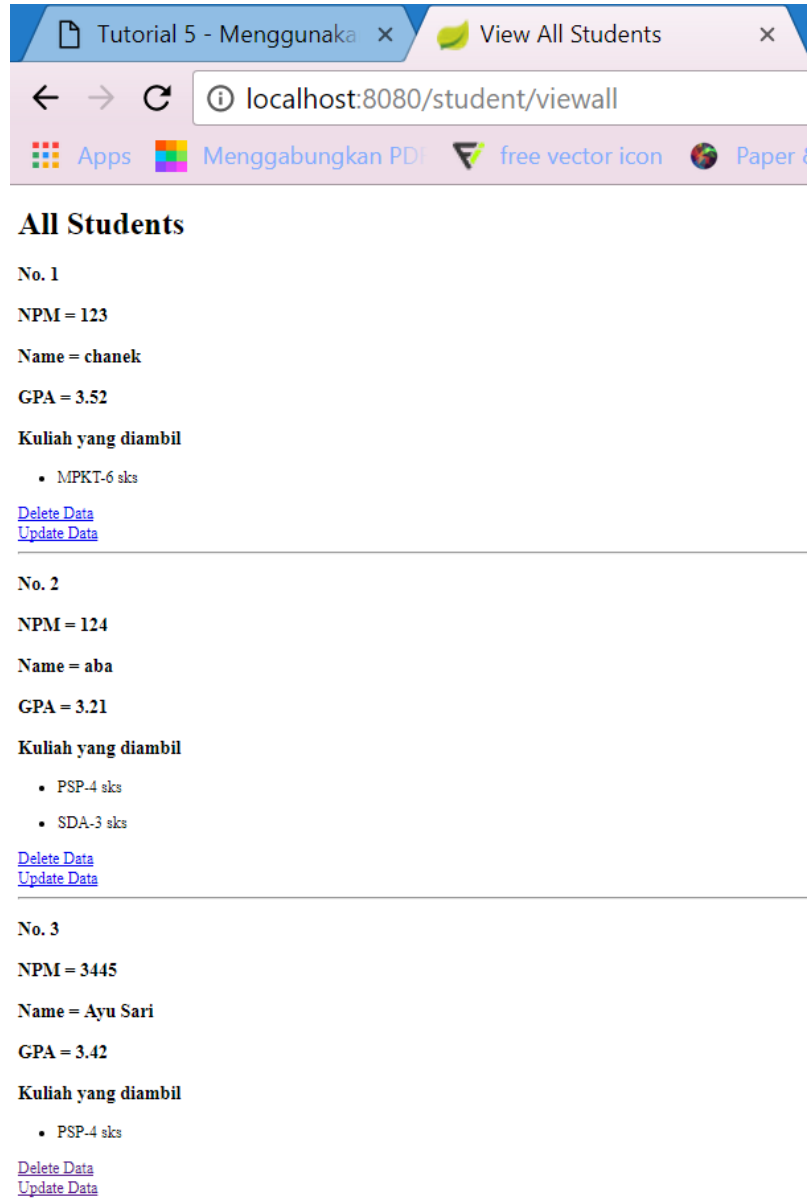
        <div th:each="student, iterationStatus: ${students}">
            <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>
            <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
            <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
            <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>

            <h3>Kuliah yang diambil</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>
```

Dan merubah html dengan menambahkan “Kuliah yang diambil” untuk menampilkan mata kuliah yang diambil mahasiswa tersebut.

Jalannya program:



All Students

No. 1

NPM = 123

Name = chanek

GPA = 3.52

Kuliah yang diambil

- MPKT-6 sks

[Delete Data](#)
[Update Data](#)

No. 2

NPM = 124

Name = aba

GPA = 3.21

Kuliah yang diambil

- PSP-4 sks
- SDA-3 sks

[Delete Data](#)
[Update Data](#)

No. 3

NPM = 3445

Name = Ayu Sari

GPA = 3.42

Kuliah yang diambil

- PSP-4 sks

[Delete Data](#)
[Update Data](#)

2. Buatlah view pada halaman `http://localhost:8080/course/view/{id}` untuk Course sehingga dapat menampilkan data course beserta Student yang mengambil.
 - 2.1. Membuat `viewCourse` pada `studentController` yang mengontrol saat course ditemukan dan tidak ditemukan dan adanya `RequestMapping` yang digunakan.

```

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

    if (course != null) {
        model.addAttribute ("course", course);
        return "viewCourse";
    } else {
        model.addAttribute ("course", course);
        return "courseNotFound";
    }
}

```

- 2.2. Membuat html untuk menampilkan mahasiswa yang mengambil course tersebut dan html saat course tidak ditemukan.

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <title>View Course</title>
    </head>
    <body>
        <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 Credit</h3>

        <h3>Mahasiswa yang mengambil</h3>
        <ul th:each="student, iterationStatus: ${course.students}">
            <li th:text="${student.npm} + '-' + ${student.name}">
            </li>
        </ul>
    </body>
</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_course}">ID Course</h3>
  </body>
</html>
```

2.3. Membuat query untuk memilih student yang mengambil mata kuliah tertentu pada StudentMapper

```
@Select("select student.npm, name, gpa " + "from studentcourse join student "
+ "on studentcourse.npm = student.npm " + "where studentcourse.id_course = #{id_course}")
List<StudentModel> selectStudents(@Param("id_course") String id_course);

@Select("select id_course, name, credits from course where id_course = #{id_course}")
@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="selectStudents"))
})
CourseModel selectCourse (@Param("id_course") String id_course);
```

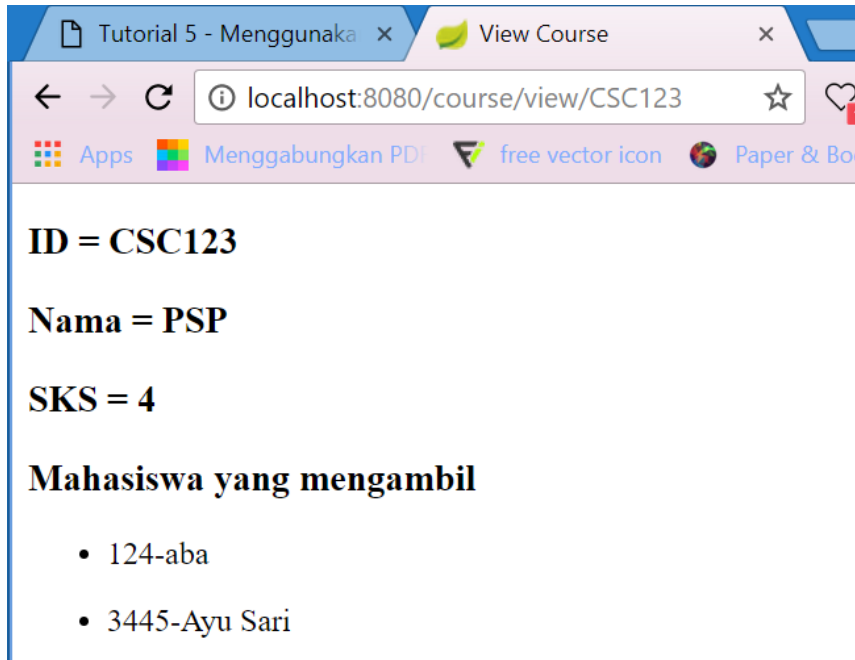
2.4. Membuat method di interface StudentService

```
CourseModel selectCourse(String idCourse);
```

2.5. Membuat method yang mengimplementasikan dari interface StudentService

```
@Override
public CourseModel selectCourse (String idCourse)
{
  Log.info ("select course");
  return studentMapper.selectCourse (idCourse);
}
```

Jalannya program:
Saat course ditemukan



Saat course yang dicari tidak ditemukan

