Ratri Ayu
1506689276
C

Tutorial 5

Menggunakan Database serta Relasi Database dalam Project Spring Boot

LATIHAN

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

   Pada method selectAllStudents di kelas StudentMapper, ditambahkan anotasi @Results untuk mengisi courses dari setiap student yang ada di List<StudentModel>.

```
@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 ();
```

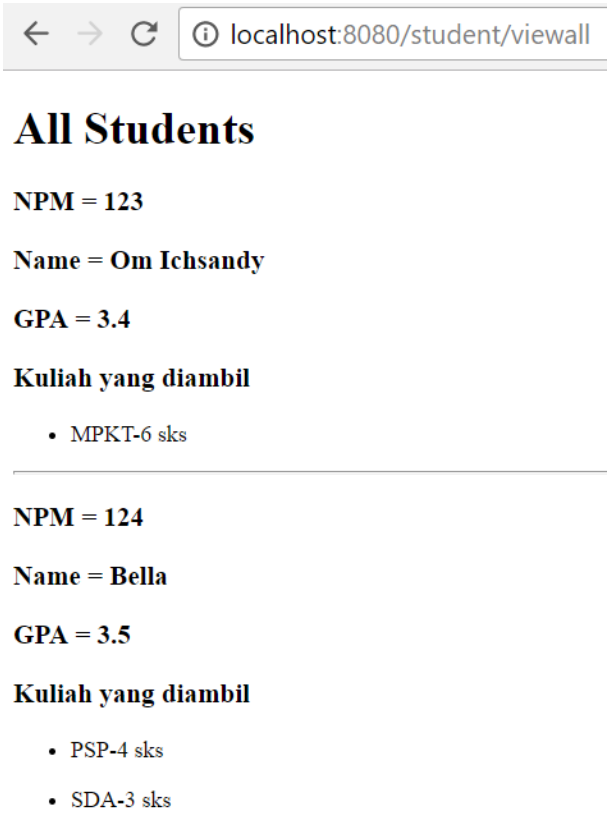
   Kemudian pada viewall.html, dilakukan iterasi courses dari setiap student.

```
<!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="'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>
            <hr/>
        </div>
    </body>
</html>
```

   Tampilan:

Ratri Ayu
1506689276
C



2. Buatlah view pada halaman http://localhost:8080/course/view/{id} untuk Course sehingga dapat menampilkan data course beserta Student yang mengambil.

   Untuk menambahkan view course, pertama dibuat method selectCourse pada StudentMapper.

```
@Select("select id_course, name, credits from course where id_course = #{id_course}")
@Results(value = {
        @Result(property = "idCourse", 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 = "selectStudentCourses"))
})
CourseModel selectCourse (@Param("id_course") String id_course);
```

   Lalu, dibuat juga method selectStudentCourse untuk mengisi List<StudentModel> dari students yang ada pada course yang dipilih.

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

Ratri Ayu
1506689276
C

Kemudian, mengimplementasi *interface* pada StudentService dan juga StudentServiceDatabase

StudentService:

```
CourseModel selectCourse (String id_course);
```

StudentServiceDatabase:

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

Pada StudentController, ditambahkan method untuk viewCourse

```
@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 {
        model.addAttribute ("course", course);
        return "CourseNotFound";
    }
}
```

Kemudian, membuat viewCourse.html dan dilakukan iterasi untuk student yang mengambil course yang dipilih

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <title>View Course by ID</title>
    </head>
    <body>
        <h3 th:text="'ID = ' + ${course.idCourse}">ID Course</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 = "student, iterationStatus: ${course.students}">
            <li th:text="${student.npm} + ' - ' + ${student.name}" >
                NPM - Nama Student
            </li>
        </ul>
    </body>
</html>
```

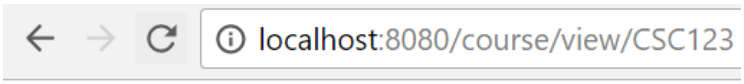
Dibuat juga CourseNotFound.html jika course tidak ditemukan

Ratri Ayu
1506689276
C

```
<!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>
```

Tampilan:

Jika course ditemukan



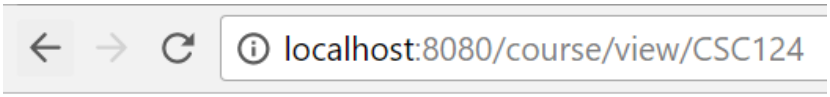
ID = CSC123

Nama = PSP

SKS = 4

Mahasiswa yang mengambil

- 124 - Bella



ID = CSC124

Nama = SDA

SKS = 3

Mahasiswa yang mengambil

- 124 - Bella

Jika course tidak ditemukan

Ratri Ayu
1506689276
C

Course not found

Course = CSC127

Ringkasan apa yang telah dipelajari:

Pada tutorial 5 ini, saya telah mempelajari mengenai pemetaan *query* pada class Model yang ada di MVC, yaitu dengan mengisi nama variabel yang dituju pada *property* dan *query* dari *database* pada *column*. Kemudian, saya juga telah mempelajari mengenai penggunaan *database* relasi pada *database* dalam project SpringBoot. Selain itu, saya juga telah mengetahui bagaimana penggunaan MyBatis dan Lombok *framework*.