Kevin Noel Anugrah
1506689370

Menggunakan Database serta Relasi Database dalam Project Spring Boot

Pada tutorial kali ini diajarkan bagaimana menggunakan spring boot dengan database yang lebih kompleks, mulai dari model, controller, dan view.

I. Menampilkan semua student dan mata kuliah yang diambil

Tambahkan anotasi result seperti selectStudent tanpa menspesifikasi npm.

```
@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 ();
```

Ubah viewall.html agar dapat menampilkan list courses.

```
<!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/update/' + ${student.npm}">Update Data</a>
                <br/><br/>
                <a th:href="'/student/delete/' + ${student.npm}">Delete Data</a>
                <hr/>
                <br/>
        </div>
    </body>
</html>
```

II. Menambahkan Course View

Buat method selectCourse(id) dan selectStudents(id) yang mirip dengan selectStudent(npm) di awal, hanya saja ganti dengan aribut – atribut courseModel.

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

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

Buat viewcourse.html yang mirip dengan view.html, hanya saja posisi course dan student ditukar.

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

Buat juga mapping nya di controller.

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