

**Sum Up yang dipelajari pada Tutorial 5:** Hal yang saya pelajari pada Tutorial 5 adalah beberapa hal sebagai berikut – yang pertama yaitu menambahkan *table* di dalam database yang sudah ada di phpmyadmin, mempopulasikan *table* tersebut, menambahkan *entity* baru, menambahkan *model*, dan hal lainnya.

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

Pada file StudentMapper.java:

```
public interface StudentMapper {  
  
    @Select("select npm, name, gpa from student where npm = #{npm}")  
    @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"))  
    })  
    StudentModel selectStudent(@Param("npm") String 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();  
  
    @Insert("INSERT INTO student (npm, name, gpa) VALUES (#{npm}, #{name}, #{gpa})")  
    void addStudent(StudentModel student);  
  
    @Delete("DELETE FROM student WHERE npm=#{npm}")  
    void deleteStudent(String npm);  
  
    @Update("UPDATE student SET name=#{name}, gpa=#{gpa} WHERE npm=#{npm}")  
    void updateStudent(StudentModel student);  
  
    @Select("select course.id_course, name, credits " +  
            "from studentcourse join course " +  
            "on studentcourse.id_course = course.id_course " +  
            "where studentcourse.npm = #{npm}")  
    List<CourseModel> selectCourses(@Param("npm") String npm);  
}
```

Melakukan perubahan pada method selectAllStudents, selectStudent. dan selectCourse untuk menampilkan hasil

Pada file viewall.html:

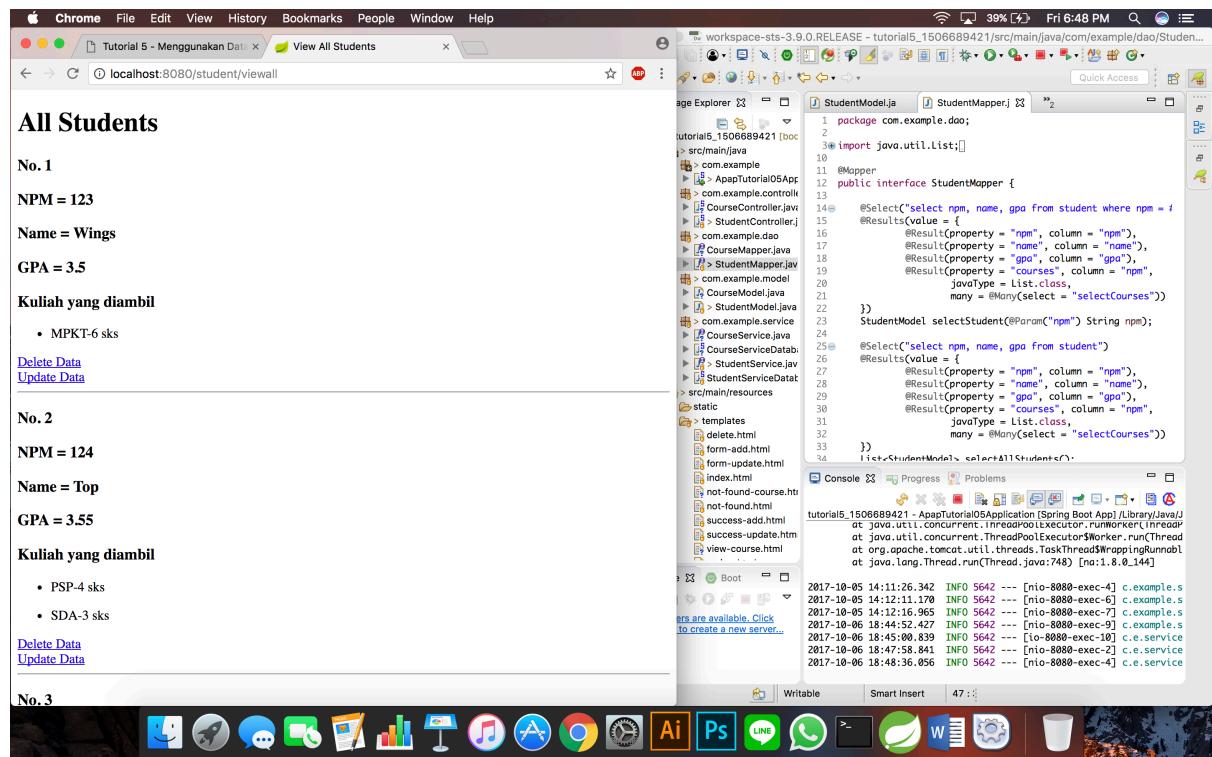
```
<div th:each="student,iterationStatus: ${students}">
    <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>
    <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
    <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
    <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA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

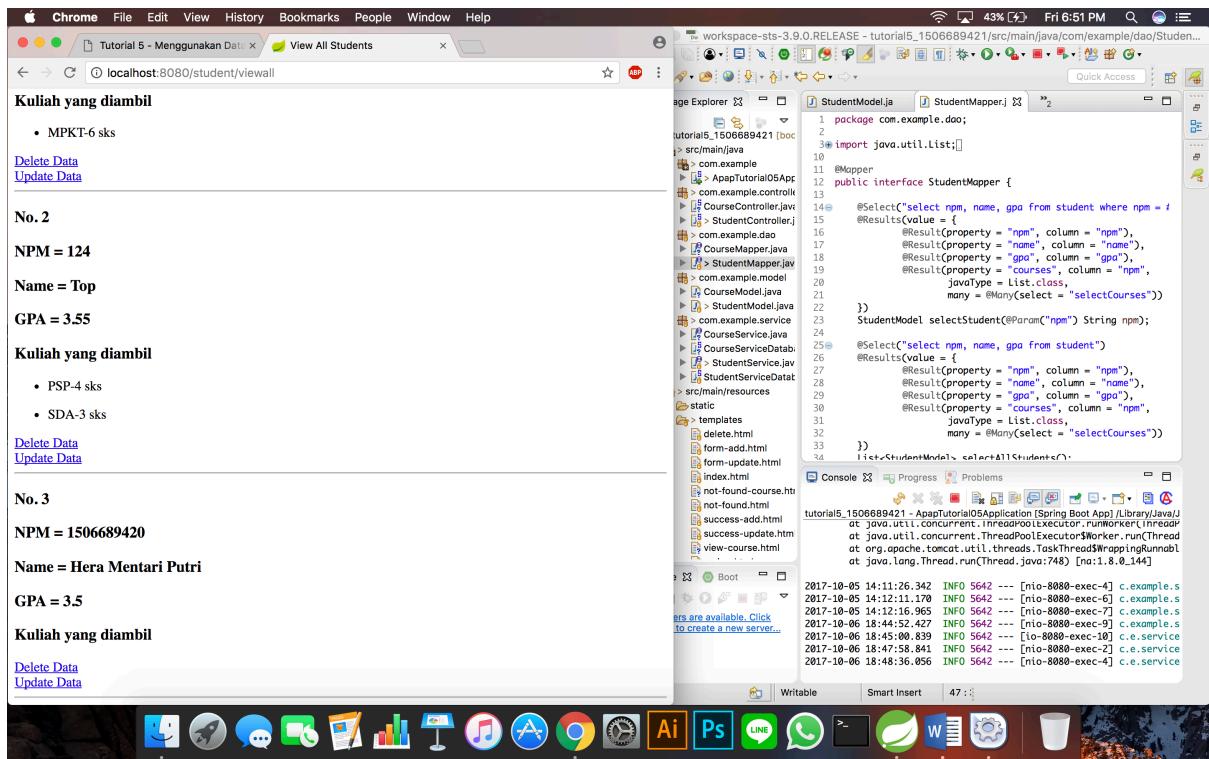
    <h3>Kuliah yang diambil</h3>
    <ul th:each="course,iterationStatus: ${student.courses}">
        <li th:text="${course.name} - ${course.credits} sks">
            Nama kuliah-X SKS</li>
    </ul>

    <a th:href="/student/delete/" +${student.npm}">Delete Data</a><br />
    <a th:href="/student/update/" +${student.npm}">Update Data</a><br />
    <hr />

</div>
```

Apabila membuka halaman <http://localhost:8080/student/viewall>, akan muncul sebagai berikut:





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

Pada file CourseController.java:

```
public class CourseController {
    @Autowired
    CourseService courseDAO;

    @RequestMapping("/course/view/{id}")
    public String viewPath(Model model, @PathVariable(value = "id") String id_course)
    {
        CourseModel course = courseDAO.selectCourse(id_course);

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

    @RequestMapping("/course/viewall")
    public String view(Model model) {
        List<CourseModel> courses = courseDAO.selectAllCourses();
        model.addAttribute("courses", courses);

        return "viewall-courses";
    }
}
```

Pada file CourseMapper.java:

```
@Mapper
public interface CourseMapper {

    @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 = "selectStudents"))
    })
    CourseModel selectCourse(@Param("id_course") String id_course);

    @Select("select id_course, name, credits from 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 = "selectStudents"))
    })
    List<CourseModel> selectAllCourses();

    @Select("select student.npm as 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);
}

}
```

Pada file CourseService.java:

```
public interface CourseService
{
    CourseModel selectCourse (String id_course);

    List<CourseModel> selectAllCourses ();
}
```

Pada file CourseServiceDatabase.java:

```
public class CourseServiceDatabase implements CourseService
{
    @Autowired
    private CourseMapper courseMapper;

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

```

@Override
public List<CourseModel> selectAllCourses ()
{
    log.info ("select all courses");
    return courseMapper.selectAllCourses ();
}

}

```

Saya juga membuat halaman untuk menampilkannya dengan sebagai berikut:

Pada file not-found-course.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}">Course ID</h3>
</body>
</html>

```

Pada file view-course.html:

```

<!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="'Name = ' + ${course.name}">Student Name</h3>
    <h3 th:text="'GPA = ' + ${course.credits}">Credits</h3>

    <h3>Mahasiswa yang mengambil</h3>
    <ul th:each="student,iterationStatus: ${course.students}">
        <li th:text="${student.npm} - ${student.name}">
            NPM - Nama mahasiswa</li>
    </ul>
</body>
</html>

```

Pada file viewall-course.html:

```

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

    <div th:each="course,iterationStatus: ${courses}">
        <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>

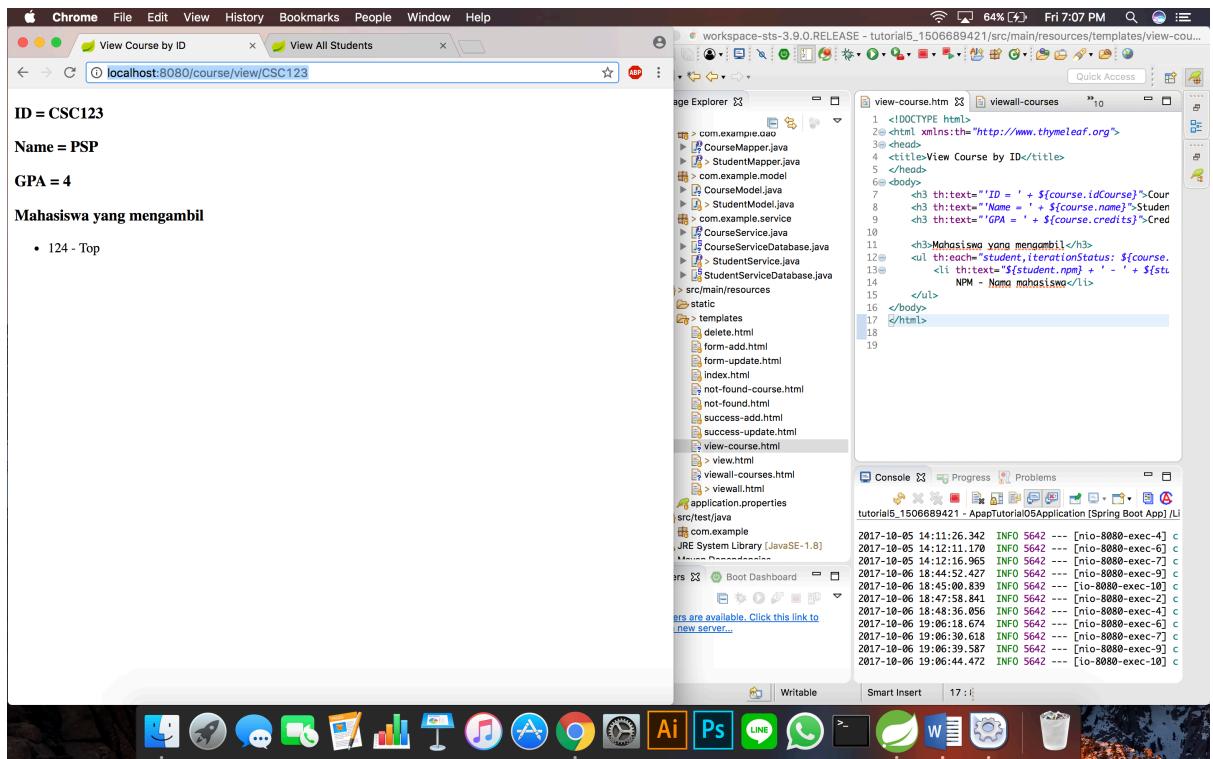
```

```
<h3 th:text="'id course = ' + ${course.idCourse}">Course ID</h3>
<h3 th:text="'Name = ' + ${course.name}">Course Name</h3>
<h3 th:text="'Credits = ' + ${course.credits}">Course Credits</h3>

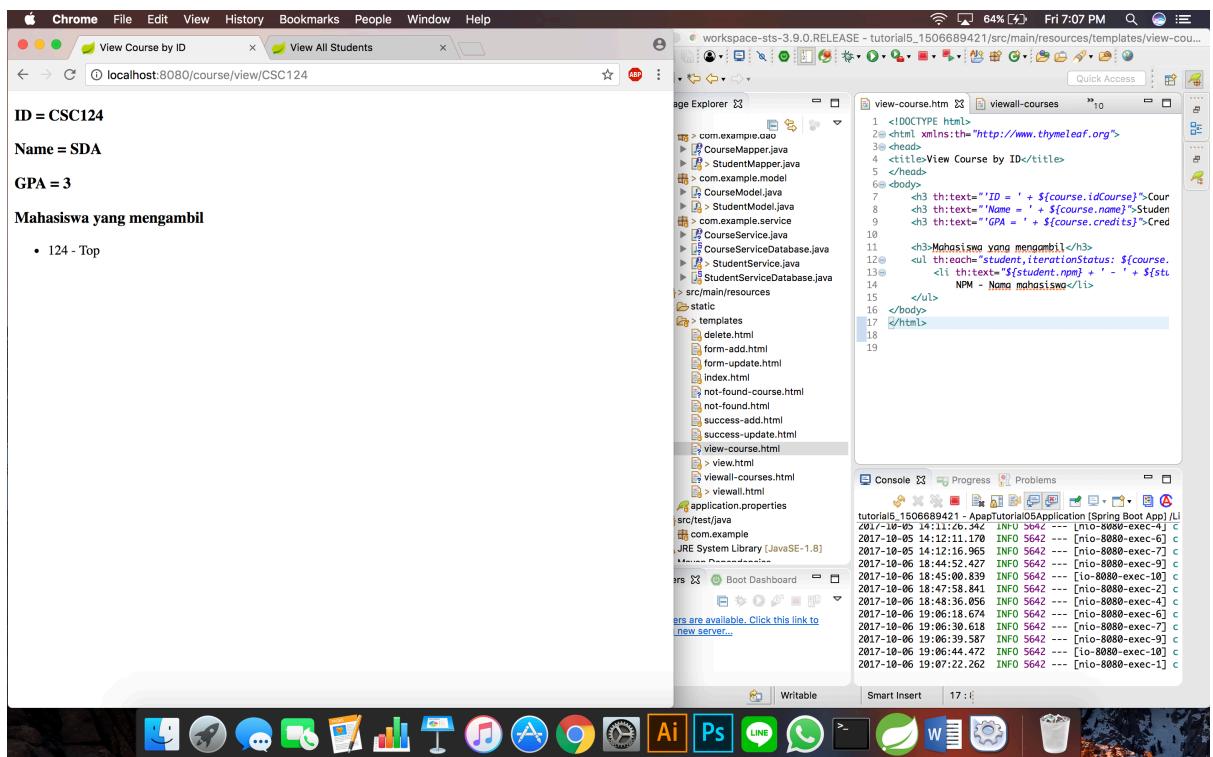
<h3>Mahasiswa yang mengambil</h3>
<ul th:each="student,iterationStatus: ${course.students}">
    <li th:text="${student.npm} - ' + ${student.name}">
        NPM - Nama mahasiswa</li>
</ul>
<hr />

</div>
</body>
</html>
```

Apabila membuka laman <http://localhost:8080/course/view/CSC123>:



Apabila membuka laman <http://localhost:8080/course/view/CSC124>:



Lalu, pengguna juga dapat melihat seluruh *course* dan siapa yang mengambilnya dengan membuka laman <http://localhost:8080/course/viewall>:

The screenshot shows the Eclipse IDE interface with the following details:

- Top Bar:** Chrome, File, Edit, View, History, Bookmarks, People, Window, Help.
- Toolbar:** Standard Eclipse icons.
- Left Sidebar:** Package Explorer, showing project structure with Java classes like CourseMapper.java, StudentMapper.java, CourseModel.java, StudentModel.java, and various Service and Database classes.
- Middle Area:**
  - View All Courses:** A list of courses:
    - No. 1: id course = CSC123, Name = PSP, Credits = 4. Mahasiswa yang mengambil: 124 - Top.
    - No. 2: id course = CSC124, Name = SDA, Credits = 3. Mahasiswa yang mengambil: 124 - Top.
    - No. 3: id course = CSC125, Name = DDP 1, Credits = 4. Mahasiswa yang mengambil: 124 - Top.
  - Code Editor:** Shows the Thymeleaf template `view-course.html` which iterates over courses and displays student names and NPMs.
  - Console:** Shows log messages from the Spring Boot application, including INFO logs for each course iteration.

This screenshot is similar to the first one but shows different student data for the same courses:

- No. 2:** id course = CSC124, Name = SDA, Credits = 3. Mahasiswa yang mengambil: 124 - Top.
- No. 3:** id course = CSC125, Name = DDP 1, Credits = 4. Mahasiswa yang mengambil: 124 - Top.
- No. 4:** id course = CSC126, Name = MPKT, Credits = 6. Mahasiswa yang mengambil: 123 - Wings.

The code editor and console output remain the same, showing the Thymeleaf template and its execution logs.