==========
Emma Sharahwati
1406557466
APAP-B
==========

Menambahkan Model

1. Menambahkan class **CourseModel** pada package com.example.model

StudentMapper.j | StudentControll | CourseModel.jav | view.html | APAPTutorial04/ | 57

```
1  package com.example.model;
2
3  import java.util.List;
4
5  import lombok.AllArgsConstructor;
6  import lombok.Data;
7  import lombok.NoArgsConstructor;
8
9  @Data
10 @AllArgsConstructor
11 @NoArgsConstructor
12 public class CourseModel {
13     private String idCourse;
14     private String name;
15     private Integer credits;
16     private List<StudentModel> students;
17 }
18
```

2. Karena Student memiliki daftar Course yang dia ambil, maka pada class **StudentModel** perlu ditambahkan list of CourseModel pada variabel:

StudentMapper.j | StudentModel.ja | CourseModel.jav

```
1  package com.example.model;
2
3  import java.util.List;
8
9  @Data
10 @AllArgsConstructor
11 @NoArgsConstructor
12 public class StudentModel {
13     private String npm;
14     private String name;
15     private double gpa;
16     private List<CourseModel> courses;
17 }
18
```

3. Selanjutnya kita akan membuat method pada **StudentMapper** yang mengembalikan list of course pada Student dengan NPM tertentu. Query tersebut akan melakukan join

antara tabel studentcourse dengan course berdasarkan id_course dan difilter hanya pada student dengan NPM yang diberikan.

4. Selanjutnya kita akan menghubungkan method selectStudent dengan method selectCourses agar saat melakukan select maka variabel List<Course> juga akan terisi. Pada class **StudentMapper** terdapat method Method tersebut diubah menjadi

StudentMapper.j | StudentModel.ja | CourseModel.jav | view.html | APAPTutorial04/ | 57

```
1  package com.example.dao;
2
3  import java.util.List;
4
5  import org.apache.ibatis.annotations.Delete;
6  import org.apache.ibatis.annotations.Insert;
7  import org.apache.ibatis.annotations.Mapper;
8  import org.apache.ibatis.annotations.Param;
9  import org.apache.ibatis.annotations.Results;
10 import org.apache.ibatis.annotations.Result;
11 import org.apache.ibatis.annotations.Select;
12 import org.apache.ibatis.annotations.Update;
13 import org.apache.ibatis.annotations.Many
14
15 import com.example.model.CourseModel;
16 import com.example.model.StudentModel;
17
18 @Mapper
19 public interface StudentMapper {
20     @Select("select   npm,   name,   gpa   from   student   where   npm   =   #{npm}")
21     @Results(value = { @Result(property = "npm", column = "npm"), @Result(property = "name", column = "name"),
22             @Result(property = "gpa", column = "gpa"),
23             @Result(property = "courses", column = "npm", javaType = List.class, many = @Many(select = "selectCourses"))
24     StudentModel selectStudent(@Param("npm") String npm);
25
26     @Select("select npm, name, gpa from student")
27     List<StudentModel> selectAllStudents();
28
29     @Insert("INSERT INTO student (npm, name, gpa) VALUES (#{npm}, #{name}, #{gpa})")
30     void addStudent(StudentModel student);
31
32     @Delete("DELETE FROM student where npm = #{npm}")
33     void deleteStudent(StudentModel student);
34
35     @Update("UPDATE student SET name = #{name}, gpa = #{gpa} WHERE npm = #{npm}")
36     void updateStudent(StudentModel student);
37
38     @Select("select   course.id_course,   name,   credits   " + "from   studentcourse   join   course   "
39             + "on   studentcourse.id_course   =   course.id_course   " + "where   studentcourse.npm   =   #{npm}")
40     List<CourseModel> selectCourses(@Param("npm") String npm);
41 }
42
```

5. Selanjutnya kita akan menambahkan *view* pada **view.html** untuk menampilkan list kuliah yang diambil oleh mahasiswa seperti berikut:

StudentMapper.j | StudentModel.ja | CourseModel.jav | view.html | APAPTutorial04/ | 57

```
1  <!DOCTYPE html>
2  <html xmlns:th="http://www.thymeleaf.org">
3  <head>
4  <title>View Student by NPM</title>
5  </head>
6  <body>
7      <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
8      <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
9      <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
10
11     <h3>Kuliah yang diambil</h3>
12     <ul th:each="course,iterationStatus:   ${student.courses}">
13         <li
14             th:text="${course.name}   +   '-'   +   ${course.credits}   +   '   sks'">
15             Nama kuliah - X SKS</li>
16     </ul>
17 </body>
18 </html>
19
```

6. Jalankan program tersebut dan buka http://localhost:8080/student/view/123 dan http://localhost:8080/student/view/124 (nomor student disesuaikan dengan database masing-masing). Tampilannya sebagai berikut:

NPM = 123

Name = Ali Wahyu Wibisono

GPA = 3.1

Kuliah yang diambil

- MPKT-6 sks

NPM = 124

Name = Anak Orion

GPA = 3.1

Kuliah yang diambil

- PSP-4 sks
- SDA-3 sks

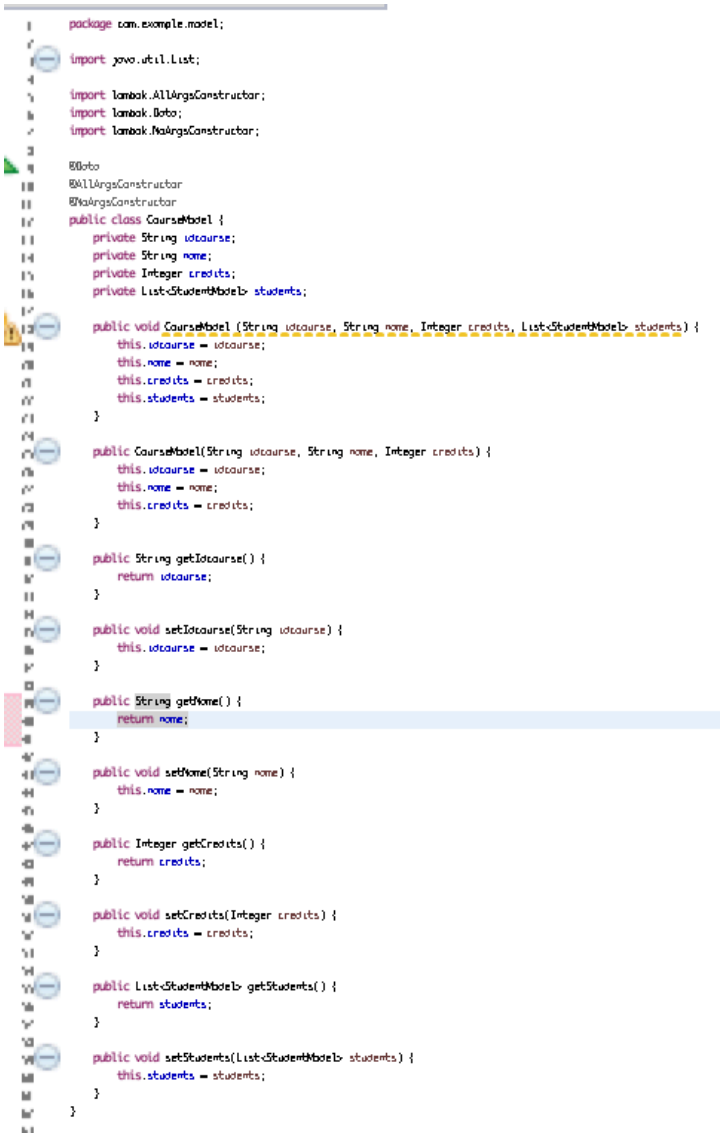
LATIHAN

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

Jawab : Saya menambahkan code berikut pada selectAllStudent pada kelas StudentMapper.

```
@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();
```

Selanjutnya, saya membuat Course Model dan melengkapi code halaman untuk viewall.

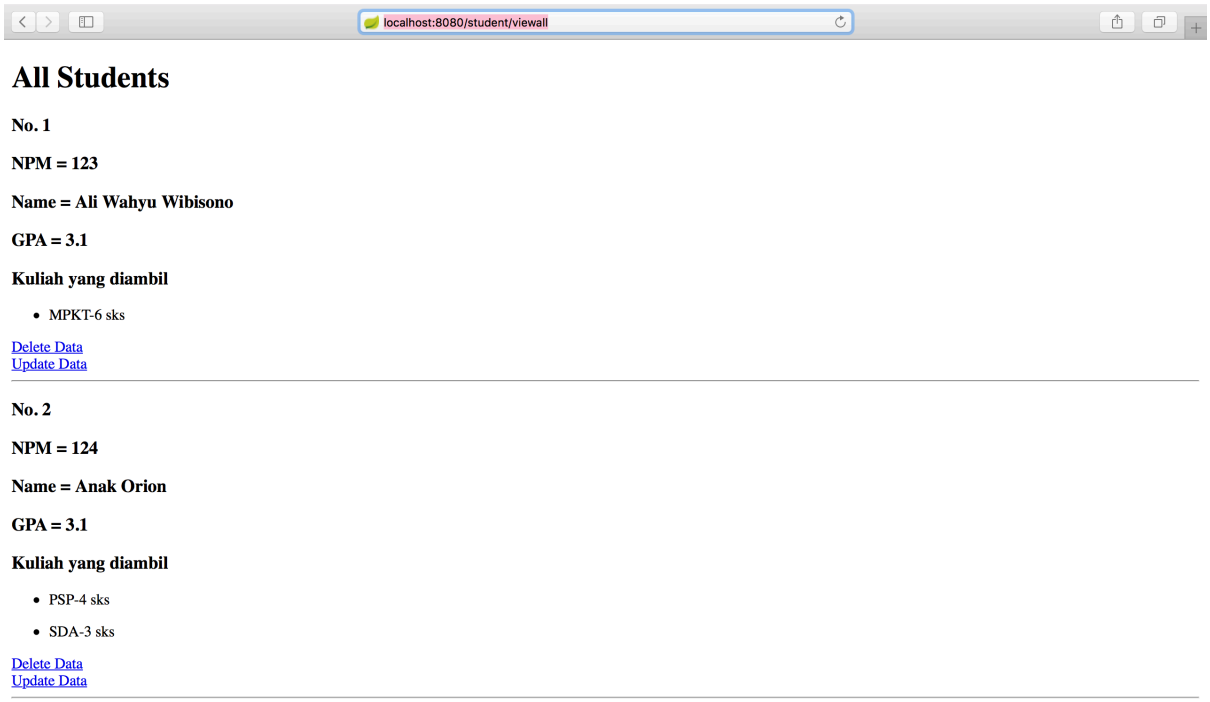


```java
package com.example.model;

import java.util.List;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@AllArgsConstructor
@NoArgsConstructor
public class CourseModel {
    private String idcourse;
    private String name;
    private Integer credits;
    private List<StudentModel> students;

    public CourseModel (String idcourse, String name, Integer credits, List<StudentModel> students) {
        this.idcourse = idcourse;
        this.name = name;
        this.credits = credits;
        this.students = students;
    }

    public CourseModel(String idcourse, String name, Integer credits) {
        this.idcourse = idcourse;
        this.name = name;
        this.credits = credits;
    }

    public String getIdcourse() {
        return idcourse;
    }

    public void setIdcourse(String idcourse) {
        this.idcourse = idcourse;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Integer getCredits() {
        return credits;
    }

    public void setCredits(Integer credits) {
        this.credits = credits;
    }

    public List<StudentModel> getStudents() {
        return students;
    }

    public void setStudents(List<StudentModel> students) {
        this.students = students;
    }
}
```

Berikut code pada halaman viewall.html yang saya buat

```
<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>
```

Code tersebut untuk menampilkan kuliah apa saja yang diambil oleh student tersebut dan hasil nya seperti berikut



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

Untuk membuat halaman untuk course, berikut langkah yang saya lakukan :

1. Membuat CourseModel seperti yang telah dijabarkan diatas.
2. Membuat Course Service

CourseModel.jav | CourseControlle | CourseService.j

```
1  package com.example.service;
2
3  import java.util.List;
4
5  import com.example.model.CourseModel;
6  import com.example.model.StudentModel;
7
8  public interface CourseService {
9      CourseModel selectCourse(String id_course);
10
11     List<CourseModel> selectAllCourses();
12 }
13
```

3. Membuat Course Service Database

StudentServiceD | CourseServiceDa | CourseModel.jav | CourseControlle | CourseService.j

```java
package com.example.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.example.dao.CourseMapper;
import com.example.model.CourseModel;
import lombok.extern.slf4j.Slf4j;
import com.example.model.StudentModel;

@Slf4j
@Service
public class CourseServiceDatabase implements CourseService {
    @Autowired
    private CourseMapper courseMapper;

    private static final org.slf4j.Logger log = org.slf4j.LoggerFactory.getLogger(CourseServiceDatabase.class);

    @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 course");
        return courseMapper.selectAllCourses();
    }

}
```

4. Membuat Course Mapper

StudentServiceD | CourseServiceDa | CourseModel.jav | CourseService.j | CourseMapper.ja

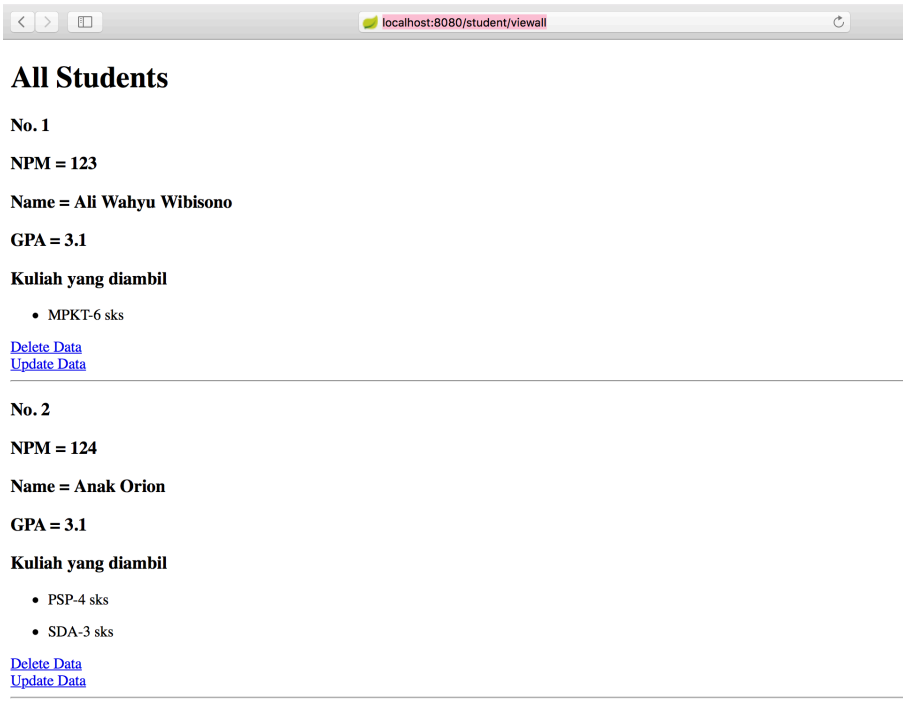
```java
package com.example.dao;

import java.util.List;

@Mapper
public interface CourseMapper {

    @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 = "idcourse", column = "id_course"),
            @Result(property = "name", column = "name"), @Result(property = "credits", column = "credits"),@Result(property = "idcourse", column = "id_course"),
            @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 = "idcourse", column = "id_course"),
            @Result(property = "students", column = "id_course", javaType = List.class, many = @Many(select = "selectStudents")) })
    List<CourseModel> selectAllCourses();


}
```

5. Membuat Course Controller

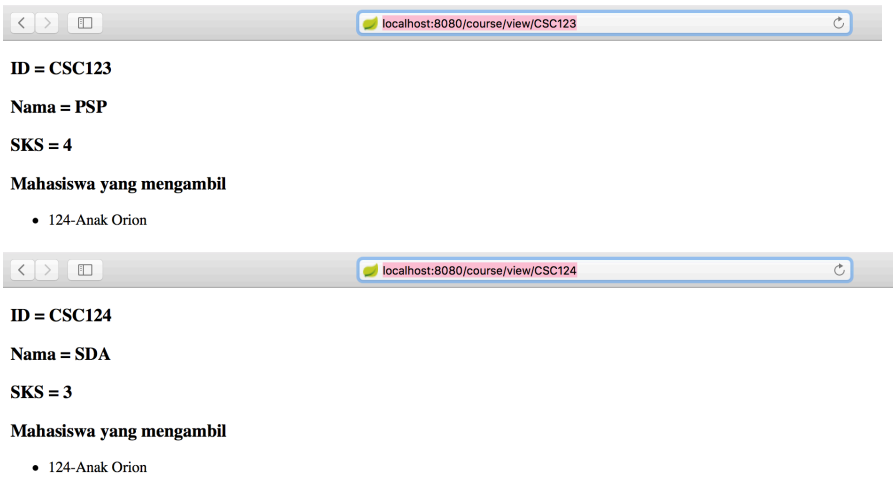
StudentServiceD | CourseServiceDa | CourseControlle

```java
package com.example.controller;

import java.util.List;

@Controller
public class CourseController {
    @Autowired
    CourseService courseDAO;

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

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

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

        return "viewall-course";
    }
}
```

6. Membuat File html untuk menampilkan course sesuai database atau inputan yang diambil. Berikut tampilan hasilnya
   - File html viewall-course.html



- File html view-course.html



- File html not-found-course.html