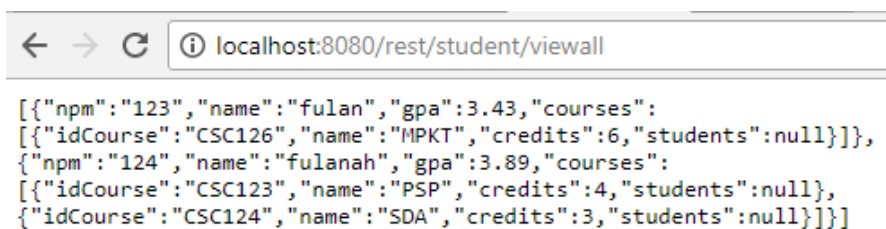


1. Latihan 1

```
@RequestMapping("/student/viewall")
public List<StudentModel> view (){
    List<StudentModel> students = studentService.selectAllStudents();
    return students;
}
```

Method tersebut akan menerima url “/rest/student/viewall”, lalu akan melakukan pengambilan data seluruh mahasiswa yang ada dengan menggunakan method yang ada pada service. Data tersebut disimpan dalam objek students dengan tipe data List<StudentModel>. Setelah data didapatkan, maka akan ditampilkan data semua student yang terdapat pada students



```
[{"npm": "123", "name": "fulan", "gpa": 3.43, "courses": [{"idCourse": "CSC126", "name": "MPKT", "credits": 6, "students": null}]}, {"npm": "124", "name": "fulanah", "gpa": 3.89, "courses": [{"idCourse": "CSC123", "name": "PSP", "credits": 4, "students": null}, {"idCourse": "CSC124", "name": "SDA", "credits": 3, "students": null}]}
```

String parse	JS eval
<pre>[{"npm": "123", "name": "fulan", "gpa": 3.43, "courses": [{"idCourse": "CSC126", "name": "MPKT", "credits": 6, "students": null}]}, {"npm": "124", "name": "fulanah", "gpa": 3.89, "courses": [{"idCourse": "CSC123", "name": "PSP", "credits": 4, "students": null}, {"idCourse": "CSC124", "name": "SDA", "credits": 3, "students": null}]}</pre>	<pre>[{"npm": "123", "name": "fulan", "gpa": 3.43, "courses": [{"idCourse": "CSC126", "name": "MPKT", "credits": 6, "students": null}]}, {"npm": "124", "name": "fulanah", "gpa": 3.89, "courses": [{"idCourse": "CSC123", "name": "PSP", "credits": 4, "students": null}, {"idCourse": "CSC124", "name": "SDA", "credits": 3, "students": null}]}</pre>

2. Latihan 2

a. Method menampilkan satu course

```
@RequestMapping("/course/view/{id}")
public CourseModel viewCourse (@PathVariable(value = "id") String id){
    CourseModel course = courseService.selectCourseInfo(id);
    return course;
}
```

Method tersebut akan menerima url “/rest/course/view” dengan variable “id”, lalu akan melakukan pengambilan data suatu course dengan id yang sesuai menggunakan method yang ada pada service. Data tersebut disimpan dalam objek course dengan tipe data CoursesModel. Setelah data didapatkan, maka akan ditampilkan data course



```
{"idCourse": "CSC123", "name": "PSP", "credits": 4, "students":
[{"npm": "124", "name": "fulanah", "gpa": 0.0, "courses": null}]}
```

String parse	JS eval
<pre>[{ "idCourse": "CSC123", "name": "PSP", "credits": 4, "students": null }, { "idCourse": "CSC124", "name": "SDA", "credits": 3, "students": null }, { "idCourse": "CSC125", "name": "DDP 1", "credits": 4, "students": null }, { "idCourse": "CSC126", "name": "MPKI", "credits": 6, "students": null }]</pre>	<pre>[{ "idCourse": "CSC123", "name": "PSP", "credits": 4, "students": null }, { "idCourse": "CSC124", "name": "SDA", "credits": 3, "students": null }, { "idCourse": "CSC125", "name": "DDP 1", "credits": 4, "students": null }, { "idCourse": "CSC126", "name": "MPKI", "credits": 6, "students": null }]</pre>

b. Method menampilkan seluruh course

```
@RequestMapping("/course/viewall")  
public List<CourseModel> viewAllCourse () {  
    List<CourseModel> courses = courseService.selectAllCourses();  
    return courses;  
}
```

Method tersebut akan menerima url `"/rest/courses/viewall"`, lalu akan melakukan pengambilan data seluruh course yang ada dengan menggunakan method yang ada pada service. Data tersebut disimpan dalam objek `courses` dengan tipe data `List<CourseModel>`. Setelah data didapatkan, maka akan ditampilkan data semua course yang terdapat pada `courses`



```
[{"idCourse": "CSC123", "name": "PSP", "credits": 4, "students": null},  
{"idCourse": "CSC124", "name": "SDA", "credits": 3, "students": null},  
{"idCourse": "CSC125", "name": "DDP 1", "credits": 4, "students": null},  
{"idCourse": "CSC126", "name": "MPKT", "credits": 6, "students": null}]
```

String parse	JS eval
<pre>{ "idCourse": "CSC123", "name": "PSP", "credits": 4, "students": [{ "npm": "124", "name": "fulanah", "gpa": 0.0, "courses": null }] }]</pre>	<pre>{ "idCourse": "CSC123", "name": "PSP", "credits": 4, "students": [{ "npm": "124", "name": "fulanah", "gpa": 0, "courses": null }] }]</pre>

3. Latihan 3

Controller akan menerima parameter “student/viewall” lalu memanggil method selectAllStudents yang terdapat pada class StudentServiceRest

```
@Override
public List<StudentModel> selectAllStudents () {
    log.info ("REST - select all students");
    return studentDAO.selectAllStudents();
}
```

Method ini kemudian akan memanggil method `selectAllStudents` yang terdapat pada class yang mengimplementasi `StudentDAO` yaitu `StudentDAOImpl`

```
@Override
public List<StudentModel> selectAllStudents(){
    List<StudentModel> students =
        restTemplate.getForObject(
            "http://localhost:8080/rest/student/viewall",
            List.class);
    return students;
}
```

Selanjutnya method ini akan mengambil informasi dari producer lalu mengembalikannya dalam objek students dengan tipe data List<StudentModel>, dimana data ini kemudian akan dibawa ke Controller kembali

Controller kemudian akan mengembalikan halaman yang berfungsi menampilkan data yang ada

localhost:9090/student/viewall

WebSiteName Home Daftar Mahasiswa Menambah Mahasiswa

All Students

No.	NPM	Name	GPA	Predicate	Delete	Update
1	123	fulan	3.43	Sangat Memuaskan!	Delete Data	Update Data
2	124	fulanah	3.89	Cum Laude!	Delete Data	Update Data

Mata Kuliah APAP

4. Latihan 4

a. Menampilkan satu course

Controller akan menerima parameter "course/view/" dengan parameter id, lalu memanggil method selectCourseInfo yang terdapat pada class CourseServiceRest

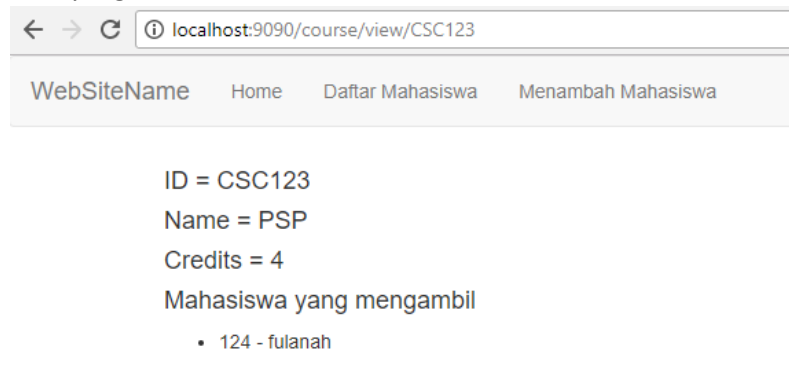
```
@Override  
public CourseModel selectCourseInfo(String id) {  
    return courseDAO.selectCourseInfo(id);  
}
```

Method ini kemudian akan memanggil method selectCourseInfo() dengan parameter id yang terdapat pada class yang mengimplementasi CourseDAO yaitu CourseDAOImpl

```
public CourseModel selectCourseInfo(String id) {  
    Log.info(id);  
    CourseModel course =  
        restTemplate.getForObject(  
            "http://localhost:8080/rest/course/view/"+id,  
            CourseModel.class);  
    return course;  
}
```

Selanjutnya method ini akan mengambil informasi dari producer lalu mengembalikannya dalam objek course dengan tipe data CourseModel, dimana data ini kemudian akan dibawa ke Controller kembali

Controller kemudian akan mengembalikan halaman yang berfungsi menampilkan data yang ada



Mata Kuliah APAP

b. Menampilkan semua course

Controller akan menerima parameter “course/viewall” lalu memanggil method selectAllCourses yang terdapat pada class CourseServiceRest

```
@Override  
public List<CourseModel> selectAllCourses(){  
    return courseDAO.selectAllCourses();  
}
```

Method ini kemudian akan memanggil method selectAllCourses() yang terdapat pada class yang mengimplementasi CourseDAO yaitu CourseDAOImpl

```
public List<CourseModel> selectAllCourses(){  
    List<CourseModel> courses =  
        restTemplate.getForObject(  
            "http://localhost:8080/rest/course/viewall",  
            List.class);  
    log.info("masuk lagi");  
    return courses;  
}
```

Selanjutnya method ini akan mengambil informasi dari producer lalu mengembalikannya dalam objek courses dengan tipe data List<CourseModel>, dimana data ini kemudian akan dibawa ke Controller kembali

Controller kemudian akan mengembalikan halaman yang berfungsi menampilkan data yang ada

← → ↻ localhost:9090/course/viewall		
WebSiteName	Home	Daftar Mahasiswa Menambah Mahasiswa
All Courses		
Course's ID	Course's Name	Credits
CSC123	PSP	4
CSC124	SDA	3
CSC125	DDP 1	4
CSC126	MPKT	6