

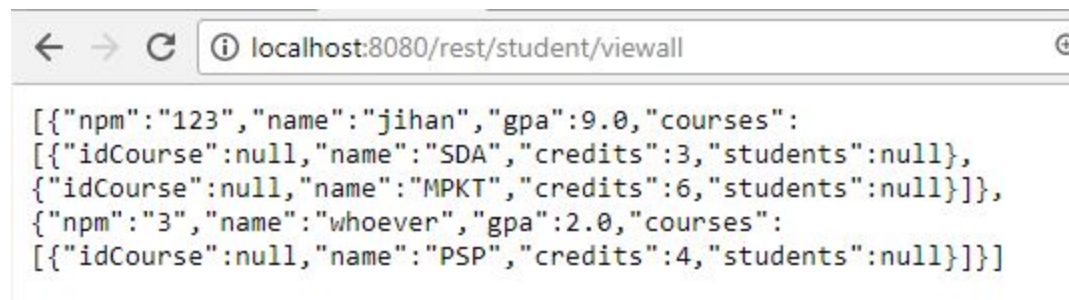
LATIHAN

- **Latihan 1: Service untuk mengembalikan seluruh student yang ada di basis data.**

Membuat method di controller yang mengembalikan list StudentModel untuk mengakses semua student yang ada di database

```
3  
6 @RequestMapping("/student/viewall")  
7 public List<StudentModel> viewall() {  
8     List<StudentModel> listStudent = studentService.selectAllStudents();  
9     return listStudent;  
0 }  
1  
2 }
```

Hasil yang ditampilkan



```
{  
  "npm": "123",  
  "name": "jihan",  
  "gpa": 9.0,  
  "courses": [  
    {  
      "idCourse": null,  
      "name": "SDA",  
      "credits": 3,  
      "students": null  
    },  
    {  
      "idCourse": null,  
      "name": "MPKT",  
      "credits": 6,  
      "students": null  
    }  
  ]  
},  
  {  
    "npm": "3",  
    "name": "whoever",  
    "gpa": 2.0,  
    "courses": [  
      {  
        "idCourse": null,  
        "name": "PSP",  
        "credits": 4,  
        "students": null  
      }  
    ]  
  }  
]
```

- **Latihan 2: Service untuk class Course.**

Membuat controller yang mengembalikan CourseModel untuk mengakses course pada database yang dicari berdasarkan ID tertentu

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

Hasil yang ditampilkan



```
{  
  "idCourse": "CSC123",  
  "name": "PSP",  
  "credits": 4,  
  "students": [  
    {  
      "npm": "3",  
      "name": "whoever",  
      "gpa": 2.0,  
      "courses": null  
    }  
  ]  
}
```

Membuat controller yang mengembalikan List of CourseModel sebagai service untuk mengakses semua course yang ada di database

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


Hasil yang ditampilkan



```
[{"idCourse": "CSC123", "name": "PSP", "credits": 4, "students": [{"npm": "3", "name": "whoever", "gpa": 2.0, "courses": null}]}, {"idCourse": "CSC124", "name": "SDA", "credits": 3, "students": [{"npm": "123", "name": "jihan", "gpa": 9.0, "courses": null}]}, {"idCourse": "CSC125", "name": "DDP 1", "credits": 4, "students": []}, {"idCourse": "CSC126", "name": "MPKT", "credits": 6, "students": [{"npm": "123", "name": "jihan", "gpa": 9.0, "courses": null}]}]
```

- **Latihan 3: Service consumer untuk view all Students**

Membuat method select all students di class studentDAOImpl untuk mengambil data List of Student Model dari producer



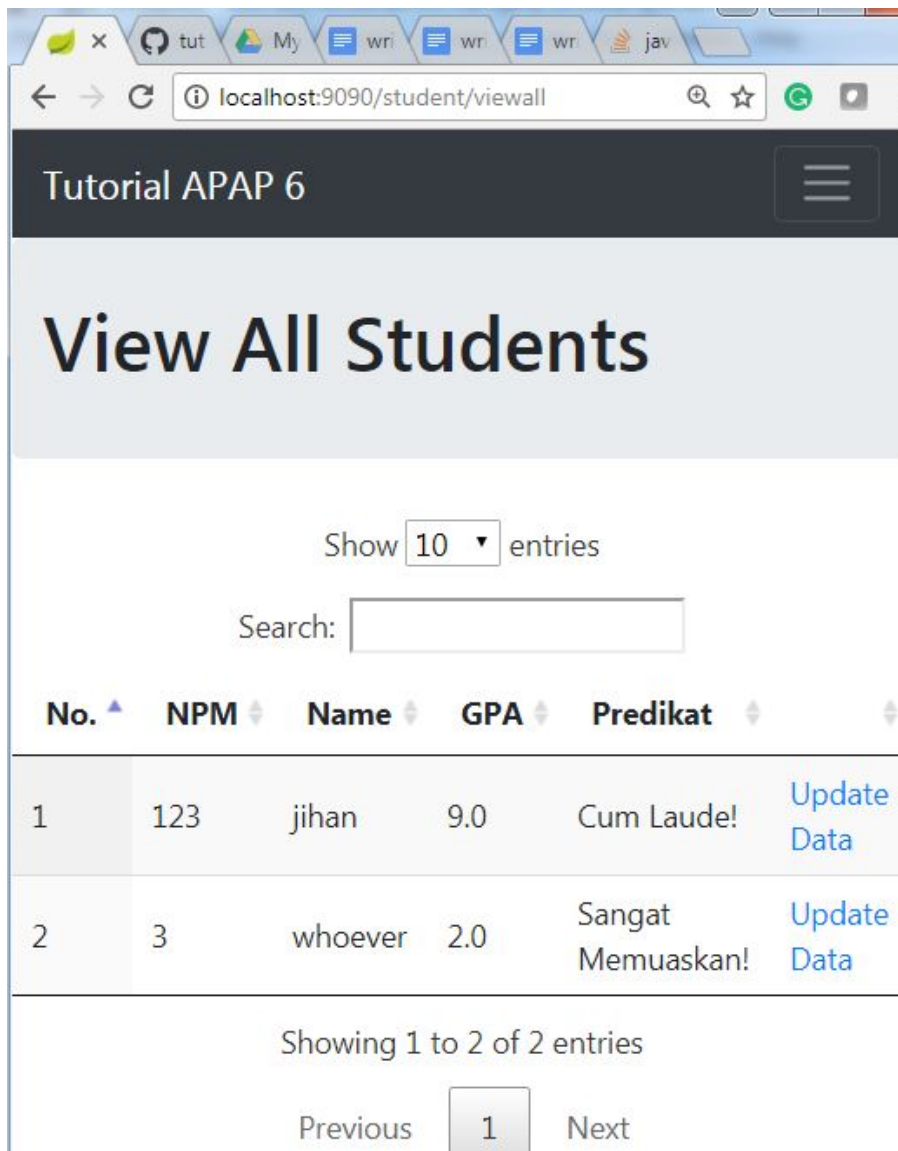
```
application.properties StudentServiceRest.java StudentDAOImpl.java
28 {
29     StudentModel student =
30         restTemplate.getForObject("http://localhost:8080/rest/student/view/"+npm, StudentModel.class);
31     return student;
32 }
33
34 @Override
35 public List<StudentModel> selectAllStudents() {
36     ResponseEntity<List<StudentModel>> rateResponse =
37         restTemplate.exchange("http://localhost:8080/rest/student/viewall",
38             HttpMethod.GET, null, new ParameterizedTypeReference<List<StudentModel>>());
39     List<StudentModel> students = rateResponse.getBody();
40     return students;
41 }
42 }
```

Membuat service dan log untuk rest

```
application.properties  StudentServiceRest.java  StudentDAO

22 @Override
23 public StudentModel selectStudent(String npm) {
24     log.info("REST - select student with npm {}", npm);
25     return studentDAO.selectStudent(npm);
26 }
27
28 @Override
29 public List<StudentModel> selectAllStudents() {
30     log.info("REST - select all students");
31     return studentDAO.selectAllStudents();
32 }
33
```

Hasil yang ditampilkan



Tutorial APAP 6

View All Students

Show 10 entries

Search:

No.	NPM	Name	GPA	Predikat	
1	123	jihan	9.0	Cum Laude!	Update Data
2	3	whoever	2.0	Sangat Memuaskan!	Update Data

Showing 1 to 2 of 2 entries

Previous 1 Next

- **Latihan 4: Service consumer untuk class CourseModel**

Membuat interface courseDAO

```
StudentDAOImpl.java CourseDAOImpl.java CourseDAO.java
1 package com.example.dao;
2
3 import java.util.List;
4
5
6
7
8 public interface CourseDAO {
9     CourseModel selectCourse(String id_course);
10    List<CourseModel> selectAllCourses();
11 }
12
```

Membuat class courseDAOimpl yang mengimplement interface di atas, dan membuat method selectAll dan selectCourse yang akan mengambil data dari producer melalui url

```
StudentDAOImpl.java CourseDAOImpl.java CourseDAO.java
9 import org.springframework.http.ResponseEntity;
10 import org.springframework.stereotype.Service;
11 import org.springframework.web.client.RestTemplate;
12
13 import com.example.model.CourseModel;
14 import com.example.model.StudentModel;
15
16 @Service
17 public class CourseDAOImpl implements CourseDAO {
18
19
20     @Autowired
21     private RestTemplate restTemplate;
22
23     @Bean
24     public RestTemplate restTemplate() {
25         return new RestTemplate();
26     }
27
28     @Override
29     public CourseModel selectCourse(String id_course) {
30
31         CourseModel course =
32             restTemplate.getForObject("http://localhost:8080/rest/course/view/"+id_course, CourseModel.class);
33         return course;
34     }
35
36     @Override
37     public List<CourseModel> selectAllCourses() {
38         ResponseEntity<List<CourseModel>> rateResponse =
39             restTemplate.exchange("http://localhost:8080/rest/course/viewall",
40                 HttpMethod.GET, null, new ParameterizedTypeReference<List<CourseModel>>() {
41                 });
42         List<CourseModel> courses = rateResponse.getBody();
43         return courses;
44     }
45 }
46 }
```


Membuat log dan service untuk rest di CourseServiceRest

```
CourseServiceRest.java
-
3 import java.util.List;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.context.annotation.Primary;
7 import org.springframework.stereotype.Service;
8
9 import com.example.dao.CourseDAO;
10 import com.example.model.CourseModel;
11
12 import lombok.extern.slf4j.Slf4j;
13
14 @Slf4j
15 @Service
16 @Primary
17 public class CourseServiceRest implements CourseService {
18
19     @Autowired
20     private CourseDAO courseDAO;
21
22     @Override
23     public CourseModel selectCourse(String id_course) {
24         log.info("REST - select course with id {}", id_course);
25         return courseDAO.selectCourse(id_course);
26     }
27
28     @Override
29     public List<CourseModel> selectAllCourses() {
30         log.info("REST - select all students");
31         return courseDAO.selectAllCourses();
32     }
33
34
35 }
36
```

Hal yang dipelajari :

Dari tutorial ini, saya mempelajari mengenai bagaimana menggunakan springboot dan database mysql untuk membuat aplikasi yang saling berkomunikasi. Aplikasi dibagi menjadi 2 bagian yaitu producer yang bertugas untuk mengakses database atau backend, dan consumer yang bertugas sebagai frontend. Komunikasi antara dua bagian ini bisa terjadi dengan RestTemplate, yang merupakan salah satu fasilitas springboot.