TUTORIAL 7 APAP

A. Membuat Service producer

1. Import berkas-berkas yang ada di tutorial 5 untuk Service producer ini.

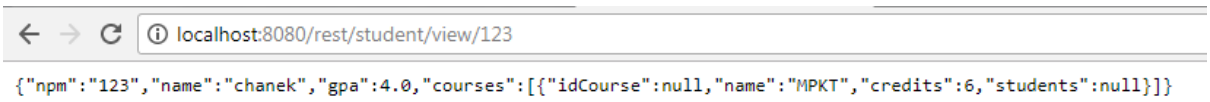
   Buatlah *project* 'Tutorial07Producer' dan silakan *import* satu *folder* atau *import file* satu per satu dari Tutorial05. Jika Anda langsung *import* satu *project* tutorial 5 jangan lupa untuk menghapus folder .git, namun jangan hapus .gitignore, dalam folder tersebut setelah dipindahkan menjadi folder Tutorial07Producer karena Anda tidak akan *push* dari direktori Tutorial07Producer.

2. Buat *package* baru com.example.rest, atau sesuaikan dengan *package* Anda masing-masing. REST Controller Web Service diletakkan pada *package* terpisah, bukan pada *package* com.example.controller

3. Buat class baru yaitu StudentRestController.java pada package com.example.rest

```java
package com.example.rest;

import java.util.List;

@RestController
@RequestMapping("/rest")
public class StudentRestController {

    @Autowired
    StudentService studentService;

    @RequestMapping("/student/view/{npm}")
    public StudentModel view(@PathVariable(value = "npm") String npm) {
        StudentModel student = studentService.selectStudent(npm);
        return student;
    }

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

    }
```

Hasilnya :



String parse


```
{
   "npm":"123",
   "name":"chanek",
   "gpa":4.0,
   "courses":[
      {
         "idCourse":null,
         "name":"MPKT",
         "credits":6,
         "students":null
      }
   ]
}
```

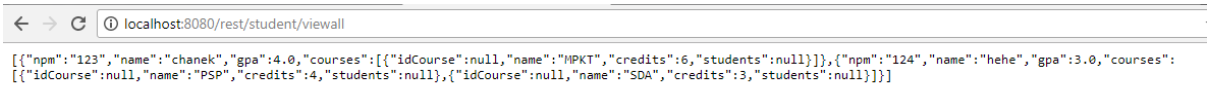

Latihan 1 :

Menambahkan method pada StudentRestService

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

}
```

Method ini untuk menampilkan semua student dengan memanggil method selectAllStudent()

Hasilnya :




```
[{"npm":"123","name":"chanek","gpa":4.0,"courses":[{"idCourse":null,"name":"MPKT","credits":6,"students":null}]},{"npm":"124","name":"hehe","gpa":3.0,"courses":
[{"idCourse":null,"name":"PSP","credits":4,"students":null},{"idCourse":null,"name":"SDA","credits":3,"students":null}]}]
```

String parse


```
[
  {
    "npm":"123",
    "name":"chanek",
    "gpa":4.0,
    "courses":[
      {
        "idCourse":null,
        "name":"MPKT",
        "credits":6,
        "students":null
      }
    ]
  },
  {
    "npm":"124",
    "name":"hehe",
    "gpa":3.0,
    "courses":[
      {
        "idCourse":null,
        "name":"PSP",
        "credits":4,
        "students":null
      },
      {
        "idCourse":null,
        "name":"SDA",
        "credits":3,
        "students":null
      }
    ]
```


Latihan 2 :

- Course/view/{id_course}

Membuat CourseRestController :

```
package com.example.rest;

import java.util.List;

@RestController
@RequestMapping("/rest")

public class CourseRestController {

    @Autowired
    StudentService studentService;

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

Method view ini nantinya akan menampilkan course berdasarkan id nya.

Hasilnya :

- Course/viewall

Karena sebelumnya tidak ada method untuk melihat semua course,maka terlebih dahulu harus membuatnya.

Membuat mapper untuk melihat semua course dan siapa saja mahasiswa yang mengambil course tersebut :

```
@Select("select * 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="joinCourse"))
})
List<CourseModel> viewAllCourse();
```

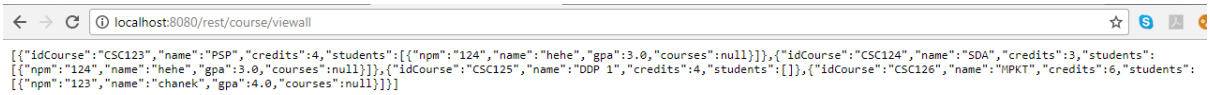
Selanjutnya,membuat method di controller untuk bisa menampilkan semua course serta mahasiswa yang mengambilnya :

```
@RequestMapping("/course/viewall")
public String viewAll (Model model)
{
    List<CourseModel> courses = studentDAO.viewAllCourse();
    model.addAttribute ("courses", courses);

    return "viewall-course";
}
```

Setelah itu,melengkapi method di CourseRestController :

```
@RequestMapping("/course/viewall")
public List<CourseModel> viewall(CourseModel model) {
    List <CourseModel> course = studentService.viewAllCourse();
    return course;

}
```

Hasilnya :



String parse


```
[
  {
    "idCourse":"CSC123",
    "name":"PSP",
    "credits":4,
    "students":[
      {
        "npm":"124",
        "name":"hehe",
        "gpa":3.0,
        "courses":null
      }
    ]
  },
  {
    "idCourse":"CSC124",
    "name":"SDA",
    "credits":3,
    "students":[
      {
        "npm":"124",
        "name":"hehe",
        "gpa":3.0,
        "courses":null
      }
    ]
  },
```

```
{
   "idCourse":"CSC125",
   "name":"DDP 1",
   "credits":4,
   "students":[
   ]
},
{
   "idCourse":"CSC126",
   "name":"MPKT",
   "credits":6,
   "students":[
      {
         "npm":"123",
         "name":"chanek",
         "gpa":4.0,
         "courses":null
      }
   ]
}
]
```

B. Membuat *Service Consumer*

1. Import berkas-berkas yang ada di tutorial 6 untuk Service Consumer ini. Buatlah *project* 'Tutorial07COnsumerdan silakan *import* satu *folder* atau *import file* satu per satu dari Tutorial06. Jika Anda langsung *import* satu *project* tutorial 6 jangan lupa untuk menghapus folder .git, namun jangan hapus .gitignore, dalam folder tersebut setelah dipindahkan menjadi folder Tutorial07Consumer karena Anda tidak akan *push* dari direktori Tutorial07Consumer.
2. Karena *Service Producer* dan *Service Consumer* harus dijalankan secara bersamaan, ubah berkas application.properties yang ada di folder resources dengan menambahkan baris

```
# Data Source
spring.datasource.platform=mysql
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/eaap
spring.datasource.username=root
spring.datasource.password=
spring.datasource.initialize=false
server.port = 9090

```

3. Di DAO tambahkan *interface* dengan nama StudentDAO
4. Isilah StudentDAO dengan kode sebagai berikut:

```
package com.example.dao;

import java.util.List;

public interface StudentDAO {

    StudentModel selectStudent(String npm);
    List<StudentModel> selectAllStudents();
}
```

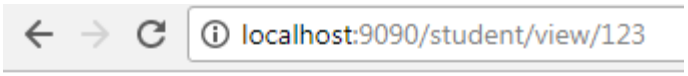
5. Selanjutnya kita akan membuat implementasi kelas StudentDAO tersebut dengan nama kelas StudentDAOImpl.java. Isinya adalah sebagai berikut

```
package com.example.dao;

import java.util.List;

@Service
public class StudentDAOImpl implements StudentDAO {

    @Autowired
    private RestTemplate restTemplate;

    @Override
    public StudentModel selectStudent(String npm) {
        StudentModel student = restTemplate.getForObject("http://localhost:8080/rest/student/view/" + npm, StudentModel.class);
        return student;
    }
```

6. Selanjutnya, kita ingin mengubah agar StudentService mengambil data dari *webservice* bukan dari *database* . Kita tidak perlu menghapus *class* StudentServiceDatabase . Karena *scalable system* , kita cukup menambahkan *class* baru yaitu StudentServiceRest yang mengimplement StudentService di *package* service.
7. Isinya adalah sebagai berikut:

```
import java.util.List;

@Slf4j
@Service
@Primary
public class StudentServiceRest implements StudentService {

    @Autowired
    private StudentDAO studentDAO;

    @Autowired
    private CourseDAO courseDAO;

    @Bean
    public RestTemplate restTemplate() {
        return new RestTemplate();
    }

    @Override
    public StudentModel selectStudent(String npm) {
        log.info("REST - Select student with npm {}" , npm);
        return studentDAO.selectStudent(npm);
    }

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

8. Jalankan kedua project Spring Boot untuk *Service Producer* dan *Service Consumer* tersebut.
9. Pastikan *service producer* sudah berjalan dengan menjalankan localhost:8080/rest/student/view/123.
10. Untuk menguji *service consumer* buka localhost:9090/student/view/123



NPM = 123

Name = chanek

GPA = 4.0

Kuliah yang diambil

- MPKT-6sks

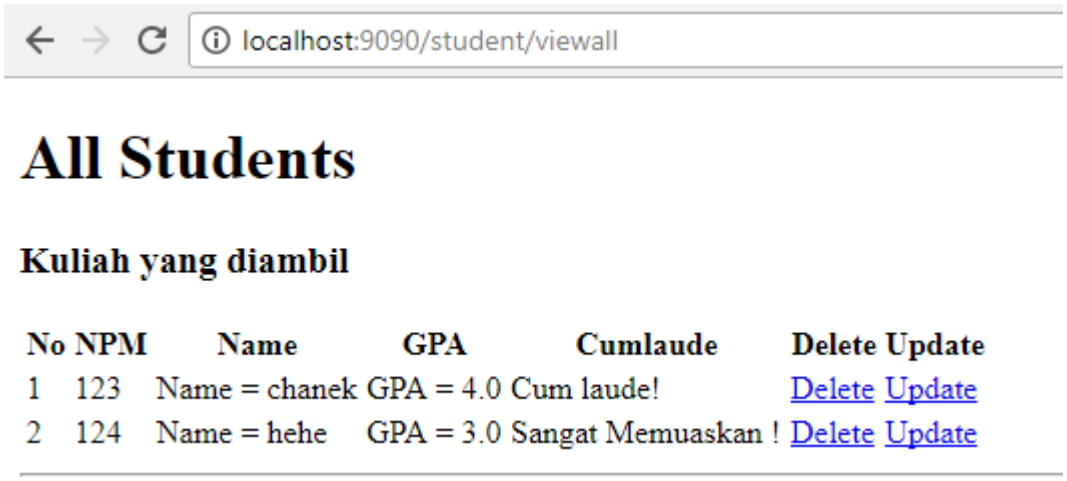
```
REST - Select student with npm 123
```

Latihan 3 :

Menambahkan method di StudentDAOImpl :

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

Hasilnya :



Latihan 4 :

Pertama-tama menambahkan interface CourseDAO dan CourseDAOImpl. Ini berisikan method-method yang nantinya digunakan untuk melihat course berdasarkan id dan melihat keseluruahn course :

```
package com.example.dao;

import java.util.List;

public interface CourseDAO {

    CourseModel viewCourse(String idCourse);
    List<CourseModel> viewAllCourse();
}
```

```java
package com.example.dao;

import java.util.List;

@Service
public class CourseDAOImpl implements CourseDAO {

    @Autowired
    private RestTemplate restTemplate;

    @Override
    public CourseModel viewCourse(String idCourse) {
        CourseModel course = restTemplate.getForObject("http://localhost:8080/rest/course/view/" + idCourse, CourseModel.class);
        return course;
    }

    @Override
    public List<CourseModel> viewAllCourse() {
        List<CourseModel> course = restTemplate.getForObject("http://localhost:8080/rest/course/viewall", List.class);
        return course;
    }

}
```

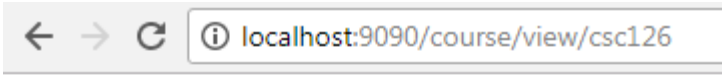
Selanjutnya adalah apa yang telah dibuat para producer seperti menambahkan mapper untuk melihat semua course dan menambahkan method pada controller untuk melihat semua course juga ditambahkan pada consumer.

Setelah itu,kita bisa mengimplementasikan method-method yang sudah dibuat pada courseDAOImpl di class StudentRestService :

```java
@Override
public CourseModel viewCourse(String idCourse) {
    log.info("REST - View Course");
    return courseDAO.viewCourse(idCourse);
}

@Override
public List<CourseModel> viewAllCourse() {
    log.info("REST - View All Course");
    return courseDAO.viewAllCourse();
}
```

Hasilnya :

(untuk view berdasarkan id)



ID = CSC126

Nama = MPKT

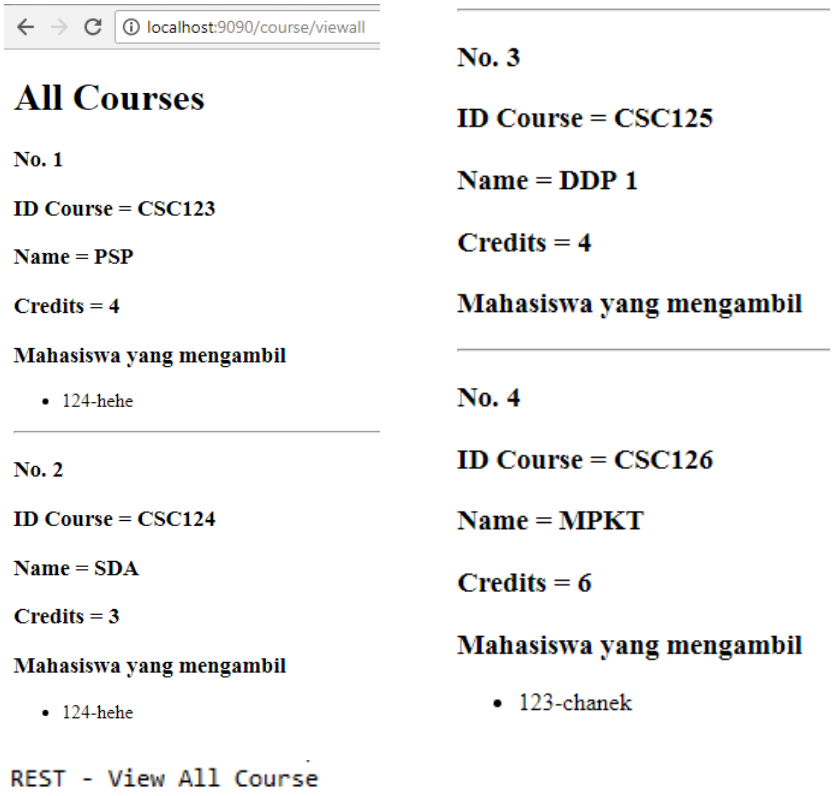
SKS = 6

Mahasiswa yang mengambil

- 123-chanek

REST - View Course

(untuk view semua course)



REST - View All Course