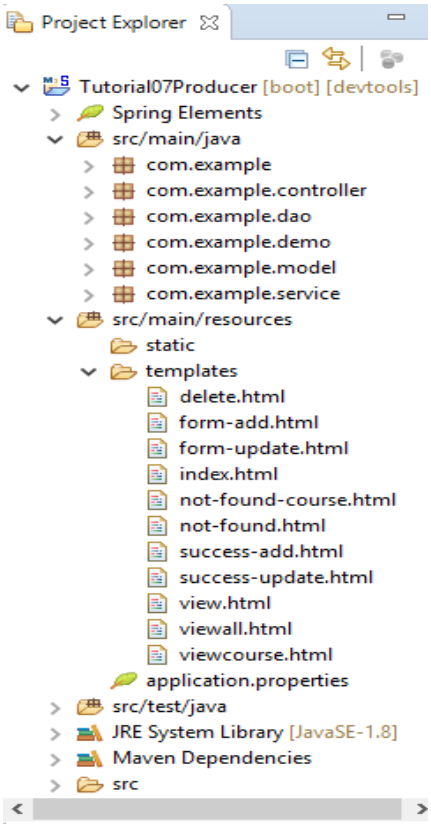
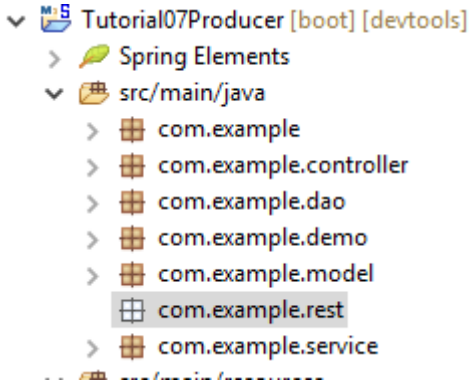
TUTORIAL 7

Membuat *Service Producer*

1. Mengimport file tutorial 5. Maka setelah di import strukturnya menjadi



2. Membuat package baru com.example.rest



3. Membuat class baru yaitu StudentRestController.java pada package yang baru saja dibuat

StudentRestController.java

```java
package com.example.rest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.model.StudentModel;
import com.example.service.StudentService;

@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;
    }
}
```

4. Localhost:8080/rest/student/view/123

localhost:8080/rest/student/view/123

Apps http://httplocalhost.c SIAKNG - Universitas Student Centered e-L


```
{"npm":"123","name":"Chanek","gpa":3.55,"courses":
[{"idCourse":"CSC126","name":"MPKT","credits":6,"students":null}]}
```


String parse


```
{
  "npm":"123",
  "name":"Chanek",
  "gpa":3.55,
  "courses":[
    {
      "idCourse":"CSC
      126",
      "name":"MPKT",
      "credits":6,
      "students":null
    }
  ]
}
```

LATIHAN

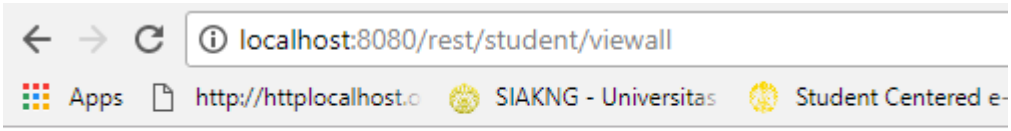
1. Latihan 1: Buatlah service untuk mengembalikan seluruh student yang ada di basis data. Service ini mirip seperti method viewAll di Web Controller. Service tersebut di-mapping ke "/rest/student/viewall".

   - Pada kelas StudentRestController membuat method baru yaitu

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

   Method ini berfungsi untuk mengambil seluruh object student yang ada dan mengembalikan list dari object seluruh student saat request mappingnya adalah /rest/student/viewall.

   - localhost:8080/rest/student/viewall




```
[{"npm":"123","name":"Chanek","gpa":3.55,"courses":
[{"idCourse":"CSC126","name":"MPKT","credits":6,"students":null}]},
{"npm":"124","name":"chanek jr.","gpa":3.4,"courses":
[{"idCourse":"CSC123","name":"PSP","credits":4,"students":null},
{"idCourse":"CSC124","name":"SDA","credits":3,"students":null},
{"idCourse":"CSC126","name":"MPKT","credits":6,"students":null}]},
{"npm":"125","name":"sav","gpa":3.8,"courses":
[{"idCourse":"CSC123","name":"PSP","credits":4,"students":null},
{"idCourse":"CSC125","name":"DDP 1","credits":4,"students":null}]}]
```

String parse

```
[
    {
        "npm":"123",
        "name":"Chanek",
        "gpa":3.55,
        "courses":[
            {
                "idCourse":"
                CSC126",
                "name":"MPKT
                ",
                "credits":6,
                "students":n
                ull
            }
        ]
    },
```

String parse JS

```
    {
        "npm":"124",
        "name":"chanek
        jr.",
        "gpa":3.4,
        "courses":[
            {
                "idCourse":"
                CSC123",
                "name":"PSP"
                ,
                "credits":4,
                "students":n
                ull
            },
            {
                "idCourse":"
                CSC124",
                "name":"SDA"
                ,
                "credits":3,
                "students":n
                ull
            },
            {
                "idCourse":"
                CSC126",
                "name":"MPKT
                ",
                "credits":6,
```

String parse J

```
    },
    {
        "npm":"125",
        "name":"sav",
        "gpa":3.8,
        "courses":[
            {
                "idCourse":"
                CSC123",
                "name":"PSP"
                ,
                "credits":4,
                "students":n
                ull
            },
            {
                "idCourse":"
                CSC125",
                "name":"DDP
                1",
                "credits":4,
                "students":n
                ull
            }
        ]
    }
]
```


2. Latihan 2: Buatlah service untuk class Course. Buatlah controller baru yang terdapat service untuk melihat suatu course dengan masukan ID Course (view by ID) dan service untuk melihat semua course (view all).

   - membuat interface CourseService pada package com.example.service yang berisi dua method berikut:

```java
package com.example.service;

import java.util.List;

import com.example.model.CourseModel;

public interface CourseService
{
    CourseModel selectCourse (String id_course);
    List<CourseModel> selectAllCourse();
}
```

Method selectCourse berfungsi untuk melihat course dengan id course tertentu dan method selectAllCourse untuk melihat seluruh course yang ada.

- membuat class CourseServiceDatabase yang mengimplementasikan CourseService

```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;

@Slf4j
@Service
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> selectAllCourse() {
        log.info("select all courses");
        return courseMapper.selectAllCourse();
    }
}
```

Kelas ini berfungsi untuk menghubungkan ke couseMapper.

- membuat courseMapper di dalam package com.example.dao

```java
package com.example.dao;

import java.util.List;

import org.apache.ibatis.annotations.Mapper;
import org.apache.ibatis.annotations.Param;
import org.apache.ibatis.annotations.Select;
import org.apache.ibatis.annotations.Result;
import org.apache.ibatis.annotations.Results;
import org.apache.ibatis.annotations.Many;

import com.example.model.CourseModel;
import com.example.model.StudentModel;

@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 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 * 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> selectAllCourse();
}
```

Interface ini berfungsi untuk mengambil data pada database yang akan dihubungkan dengan model course. SelectCourse adalah untuk mengambil data dengan id course tertentu dan memunculkan seluruh nama student yang mengambil course tersebut. Sedangkan selectAllCourse berfungsi untuk mengambil seluruh detail course yang ada pada database.

- membuat controller untuk course pada package com.example.controller

StudentRestCo... | CourseService.... | CourseService... | CourseMapper.j... | CourseControll...

```java
package com.example.controller;

import java.util.List;

@Controller
public class CourseController
{
    @Autowired
    CourseService courseDAO;

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

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

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

        return "viewall-course";
    }

}
```

- membuat html viewall-course untuk menampilkan seluruh course

CourseServic... | CourseMappe... | CourseContr... | viewall-cou... | StudentMapp

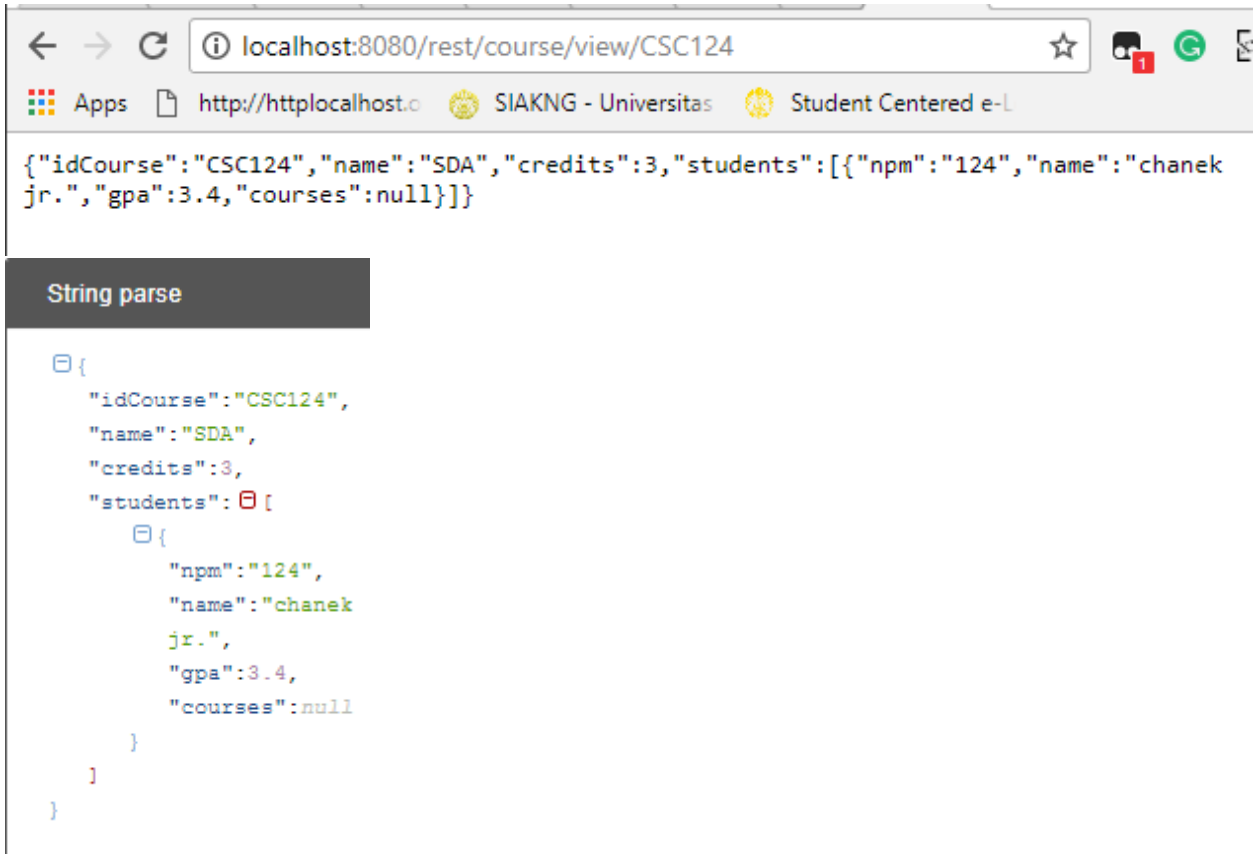
```html
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>View All course</title>
</head>
<body>
    <h1>All courses</h1>

    <div th:each="course,iterationStatus: ${courses}">
        <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>
        <h3 th:text="'ID = ' + ${course.idCourse}">course NPM</h3>
        <h3 th:text="'Name = ' + ${course.name}">course Name</h3>
        <h3 th:text="'SKS = ' + ${course.credits}">course GPA</h3>

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

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

- membuat rest controller untuk course

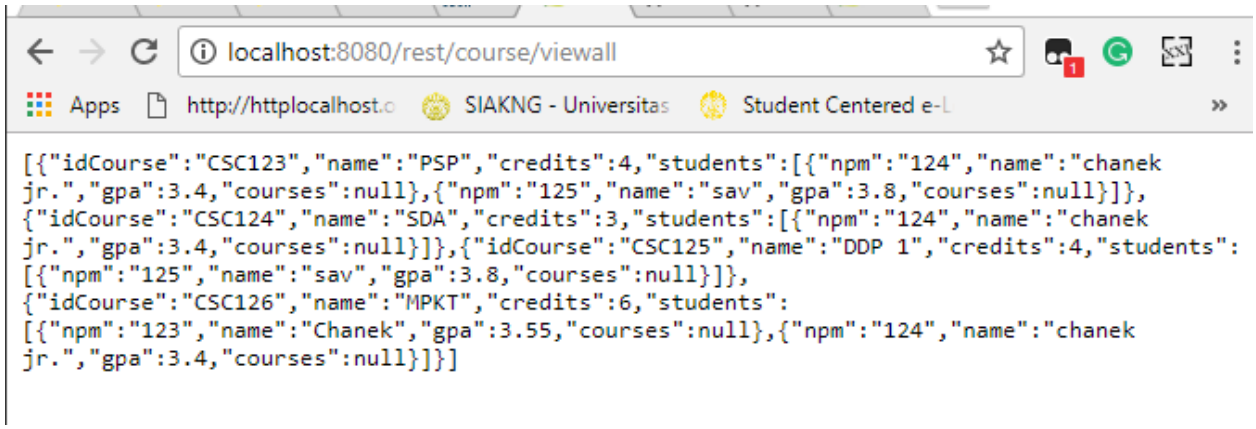
```java
package com.example.rest;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.model.CourseModel;
import com.example.service.CourseService;

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

    @Autowired
    CourseService courseService;

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

    @Autowired
    @RequestMapping("/course/viewall")
    public List<CourseModel> viewAll() {
        List<CourseModel> courses = courseService.selectAllCourse();
        return courses;
    }

}
```

Method pertama berfungsi untuk mengembalikan object course dengan id tertentu sedangkan method kedua berfungsi untuk mengembalikan seluruh object course yang ada di database. Semua object dikembalikan dalam format JSON.

- localhost:8080/rest/course/view/CSC124



- localhost:8080/rest/course/viewall

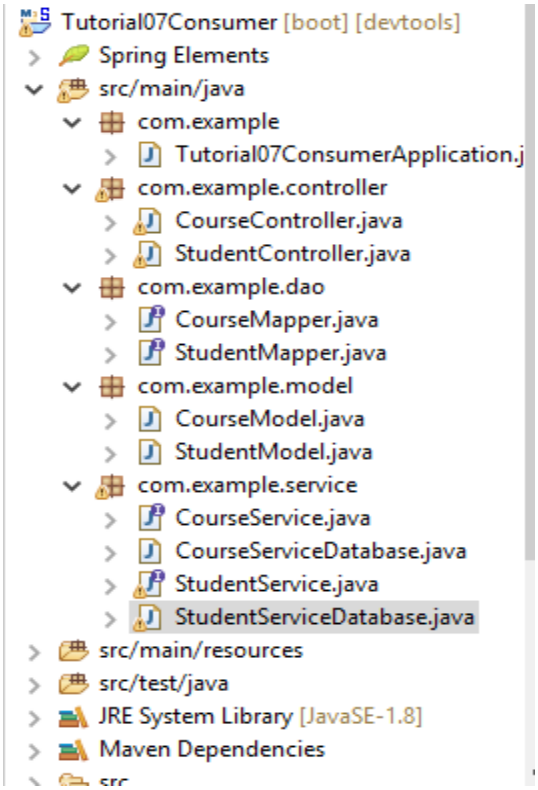


String parse

```
[
    {
        "idCourse":"CSC123
        ",
        "name":"PSP",
        "credits":4,
        "students": [
            {
                "npm":"124",
                "name":"chan
                ek jr.",
                "gpa":3.4,
                "courses":nu
                ll
            },
            {
                "npm":"125",
                "name":"sav"
                ,
                "gpa":3.8,
                "courses":nu
                ll
            }
        ]
    },
    {
        "idCourse":"CSC124
        ",
        "name":"SDA",
        "credits":3,
        "students": [
            {
                "npm":"124",
                "name":"chan
                ek jr.",
                "gpa":3.4,
                "courses":nu
                ll
            }
        ]
    },
    {
        "idCourse":"CSC125
        ",
        "name":"DDP 1",
        "credits":4,
        "students": [
            {
                "npm":"125",
                "name":"sav"
                ,
                "gpa":3.8,
                "courses":nu
                ll
            }
    },
    {
        "idCourse":"CSC126
        ",
        "name":"MPKT",
        "credits":6,
        "students": [
            {
                "npm":"123",
                "name":"Chan
                ek",
                "gpa":3.55,
                "courses":nu
                ll
            },
            {
                "npm":"124",
                "name":"chan
                ek jr.",
                "gpa":3.4,
                "courses":nu
                ll
            }
        ]
    }
]
```


Membuat *Service Consumer*

1. Mengimport tutorial 6 kedalam project Tutorial07Consumer sehingga strukturnya dalah sebagai berikut:



2. Mengubah application.properties dimana port server menjadi 9090

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

3. Pada DAO menambahkan interface dengan nama student DAO dan berisi hal berikut

```
StudentServi...   application....   StudentDAO.java
package com.example.dao;

import java.util.List;

import com.example.model.StudentModel;

public interface StudentDAO {
    StudentModel selectStudent (String npm);
    List<StudentModel> selectAllStudents();
}
```

4. Membuat implementasi StudentDAO dengan nama kelas StudentDAOImpl.java yang berisi hal berikut

```java
package com.example.dao;

import java.util.List;

@Service
public class StudentDAOImpl implements StudentDAO {
    @Autowired
    private RestTemplate restTemplate;

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

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

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

5. Menambahkan class baru StudentServiceRest di package service dengan isi sebagai berikut

```java
import com.example.dao.StudentDAO;
import com.example.model.StudentModel;

import lombok.extern.slf4j.Slf4j;

@Slf4j
@Service
@Primary
public class StudentServiceRest implements StudentService {
    @Autowired
    private StudentDAO studentDAO;

    @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();
    }

    @Override
    public void addStudent(StudentModel student) {
    }

    @Override
    public void deleteStudent(String npm) {
    }

    @Override
    public void updateStudent(StudentModel student) {
    }
}
```

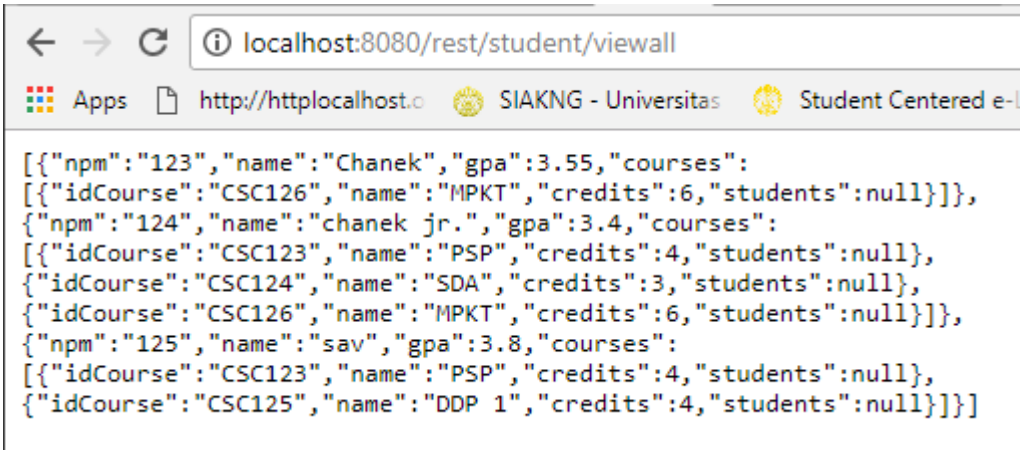
6. localhost:8080/rest/student/view/123 dan setelah itu menjalankan localhost:9090/student/view/123

LATIHAN

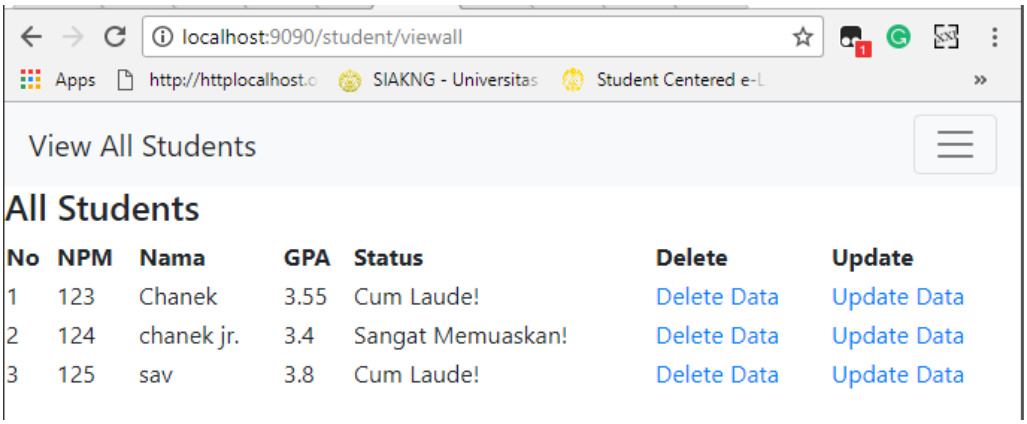
3. Latihan 3: Implementasikan service consumer untuk view all Students dengan melengkapi method selectAllStudents yang ada di kelas StudentServiceRest.
   - membuat method selectAllStudents di kelas StudentDAOImpl dengan isi berikut

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

   Dimana akan mengambil data object ketika diakses localhost:8080/rest/student/viewall dan menampilkannya sesuai dengan object yang diambil
   - menjalankan localhost:8080/rest/student/viewall



- menjalankan localhost:9090/student/viewall



4. Latihan 4: Implementasikan service consumer untuk class CourseModel dengan membuat class-class DAO dan service baru.

- membuat interface CourseDAO di package DAO

CourseDAO.java

```
1  package com.example.dao;
2
3  import java.util.List;
4
5  import com.example.model.CourseModel;
6
7  public interface CourseDAO {
8      CourseModel selectCourse (String id);
9      List<CourseModel> selectAllCourse();
10 }
```

- membuat class CourseDAOimpl yang mengimplemntasikan CourseDAO

```java
CourseDAOImpl.java   CourseServiceRest.java   StudentDAOImpl.java
package com.example.dao;

import java.util.List;

@Service
public class CourseDAOImpl implements CourseDAO {
    @Autowired
    private RestTemplate restTemplate;

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

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

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

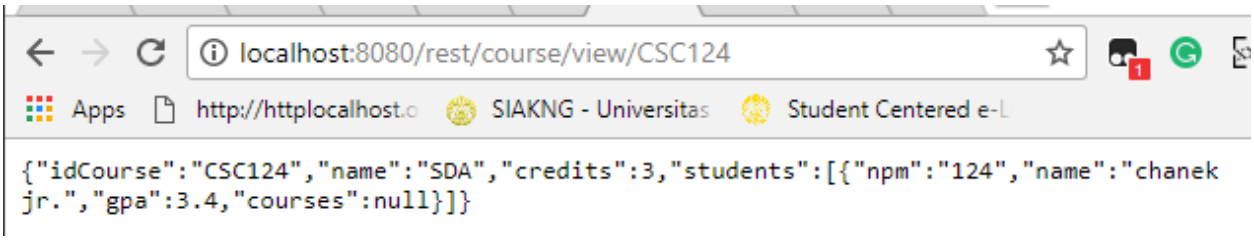
Method pertama akan mengambil data object course dengan id course tertentu setelah mengakses localhost:8080/rest/course/view/{id_course} dan method kedua akan mengambil seluruh data object course setelah mengakses localhots:8080/rest/course/viewall.

- membuat CourseServiceRest di package service

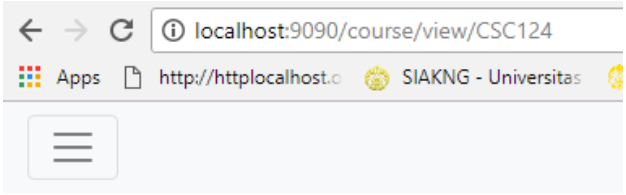
```java
CourseDAOImpl.java   CourseServiceRest.java
package com.example.service;

import java.util.List;

@Slf4j
@Service
@Primary
public class CourseServiceRest implements CourseService {
    @Autowired
    private CourseDAO courseDAO;

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

    @Override
    public List<CourseModel> selectAllCourse() {
        log.info("REST - select all course");
        return courseDAO.selectAllCourse();
    }

}
```

Student Service akan mengambil data dari web service sehingga menambahkan class baru yaitu CourseServiceRest yang mengimplemen CourseService pada package service. Selain itu, menambahkan anotasi @Primary sehingga Autowired akan secara otomatis menginstansiasi CourseService menggunakna CourseServiceRest.

- localhost:8080/rest/course/view/CS124



- localhost:9090/course/view/CS124



ID = CSC124

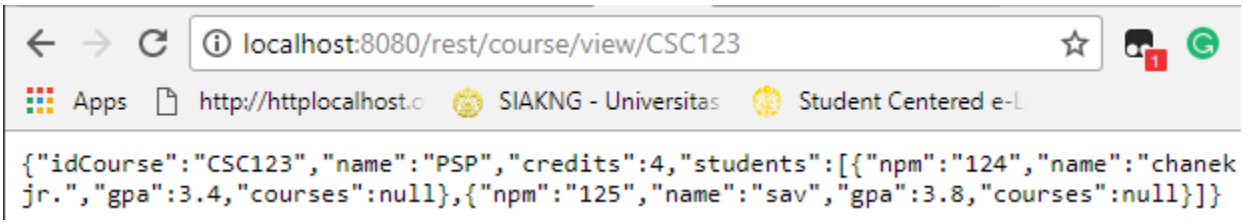
Name = SDA

Credits = 3

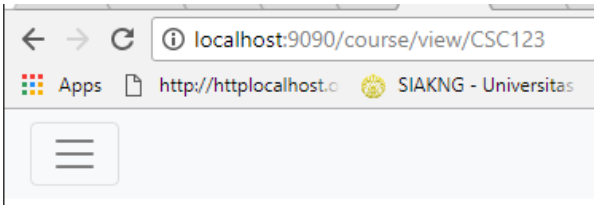
Mahasiswa yang mengambil :

- 124-chanek jr.

- localhost:8080/rest/course/view/CSC123



- localhost:9090/course/view/CSC123



ID = CSC123

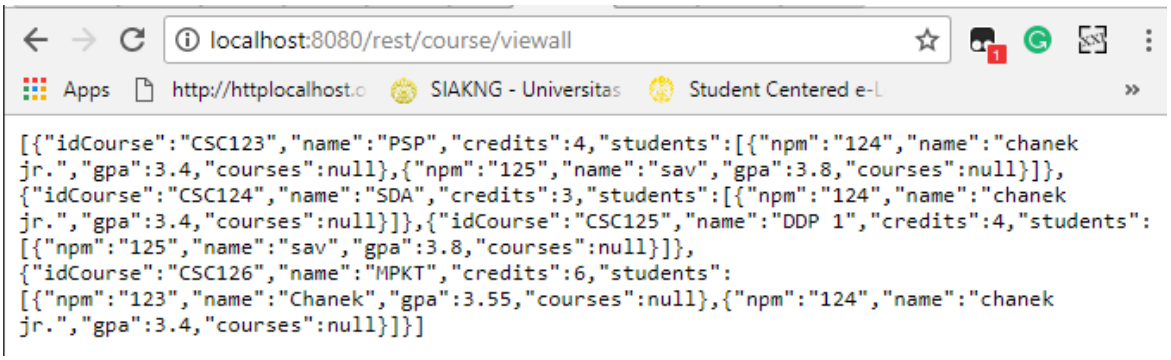
Name = PSP

Credits = 4

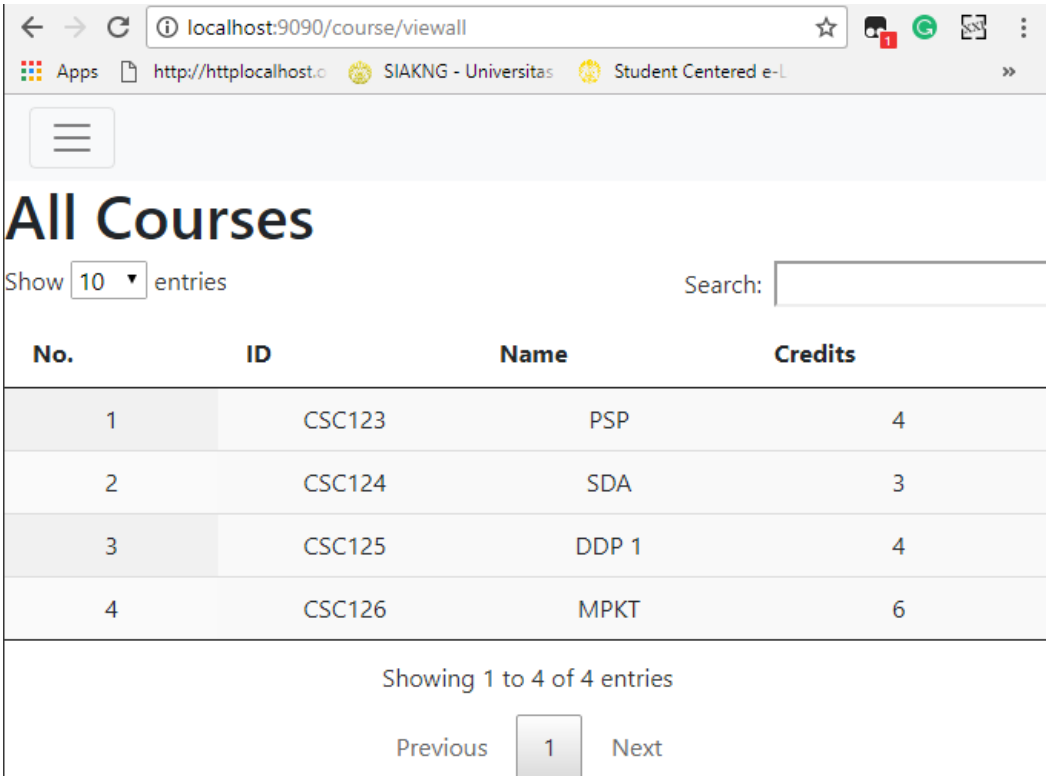
Mahasiswa yang mengambil :

- 124-chanek jr.
- 125-sav

- localhost:8080/rest/course/viewall



- localhost:9090/course/viewall



Lesson Learned

Pada tutorial7 kali ini saya belajar menegnai web service dengan menggunakan spring boot framework. Pada tutorial kali ini akan dibuat dua buah project yaitu project untuk producer dan consumer dimana terjadi pemisahan layer backend (service producer) dan frontend(service consumer). Service consumer merupakan aplikasi yang berinteraksi dengan pengguna sedangkan service producer merupakan aplikasi yang akan mengambil data yang service consumer minta serta memberikannya kepada service consumer. Selain itu, pada service producer, digunakan mapper untuk mengambil data-data objek dari database dan pada service consumer tidak menggunakan mapper atau mengambil data dari database karena ia menyediakan dan mengolah data dari service producer. Service producer biasanya juga tidak memiliki view yang dapat dilihat mengguna sedangkan service consumer memiliki view yang berfungsi untuk ditampilkan.

Untuk menghubungkan keduanya, service producer dapat menyediakan web service yang dapat dikonsumsi oleh service consumer yaitu sebuah URL yang akan mengembalikan data yang pada tutorial ini adalah JSON. Selain itu, pada service producer akan dibuat controller dengan menggunakan REST web service dan dengan anotasi @RestController dan pada class header ditambahkan @RequestMapping("/rest") agar setiap pemanggilan harus diawali dengan "rest", hal ini pada service consumer hal ini tidak dibutuhkan. Sedangkan pada service consumer, web service ditempatkan di layer DAO dan karena keduanya dijalankan secara bersamaan maka port server pada consumer dan producer harus berbeda. Nantinya akan terdapat class yang mengimplentasikan interface dari objectDAO yang kita buat. Class tersebut akan menggunakan RestTemplate yang berfungsi untuk mengkonsumsi object REST web service yaitu dengan menggunakan method getForObject yang menerima parameter berupa URL producer web service dan tipe class dari object yang didapatkan. Selain itu, akan dibuat kelas yang mengimplentasi interface service untuk mengambil data dari web service. Karena pada service consumer tidak menggunakan mapper maka pada terdapat dua implentasi untuk interface service, dimana satu memanggil mapper dan satu lagi tidak. Cara agar controller mengambil data dari web service adalah menggunakan anotasi @Primary dimana Autowired akan secara otomatis menginstansiasi interface Service menggunakan ServiceRest karena dia primary