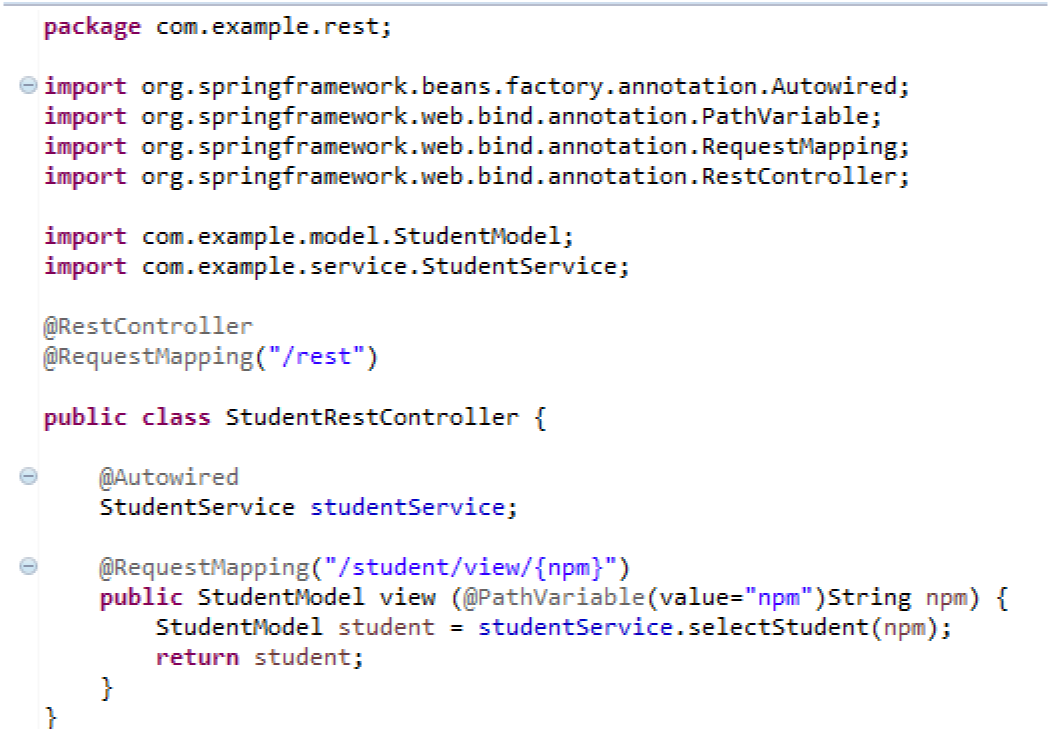
Membuat Web Service Menggunakan Spring Boot Framework

Membuat Service Procedur

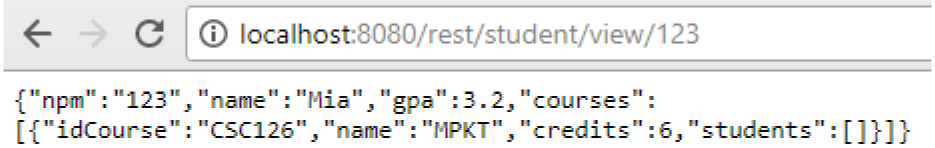
Web Service disediakan oleh *service producer* agar dapat berkomunikasi dengan *service consumer*, yang akan mengembalikan data dalam representasi seperti JSON atau XML.

1. Mengimpor berkas-berkas yang ada pada tutorial 5 untuk ServiceProcedur
2. Membuat package baru com.example.rest yang berisi StudentRestController.java sebagai berikut :



```
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;
    }
}
```

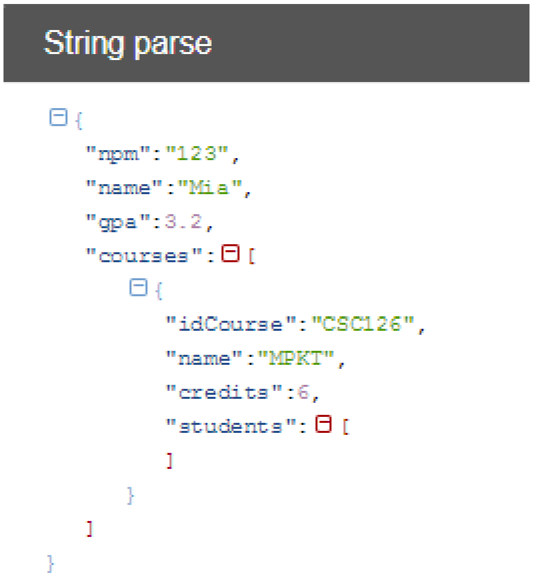
3. Mengakses **localhost:8080/rest/student/view/123**



```
{"npm":"123","name":"Mia","gpa":3.2,"courses":
[{"idCourse":"CSC126","name":"MPKT","credits":6,"students":[]}]}
```

- Method-method di REST Controller akan mengembalikan objek dengan format JSON yang ingin dikembalikan oleh service pada tampilan Web.

4. Untuk memudahkan pembacaan menggunakan JSON View dengan mengakses Hasilnya sebagai berikut :



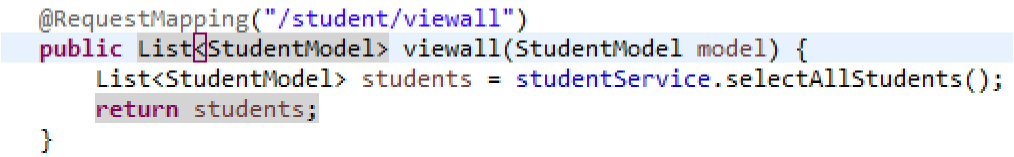
String parse

```
{
    "npm":"123",
    "name":"Mia",
    "gpa":3.2,
    "courses":[
        {
            "idCourse":"CSC126",
            "name":"MPKT",
            "credits":6,
            "students":[
            ]
        }
    ]
}
```

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".

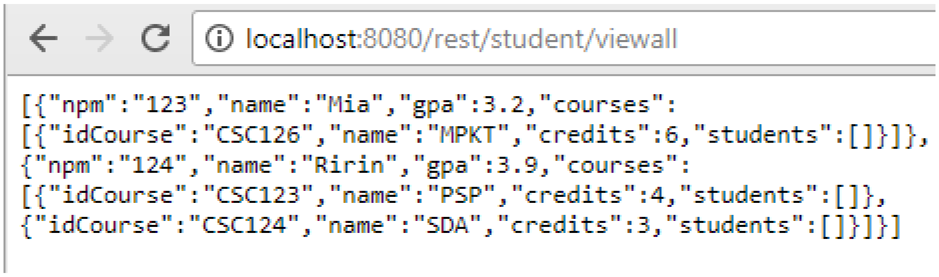
   StudentRestController.java



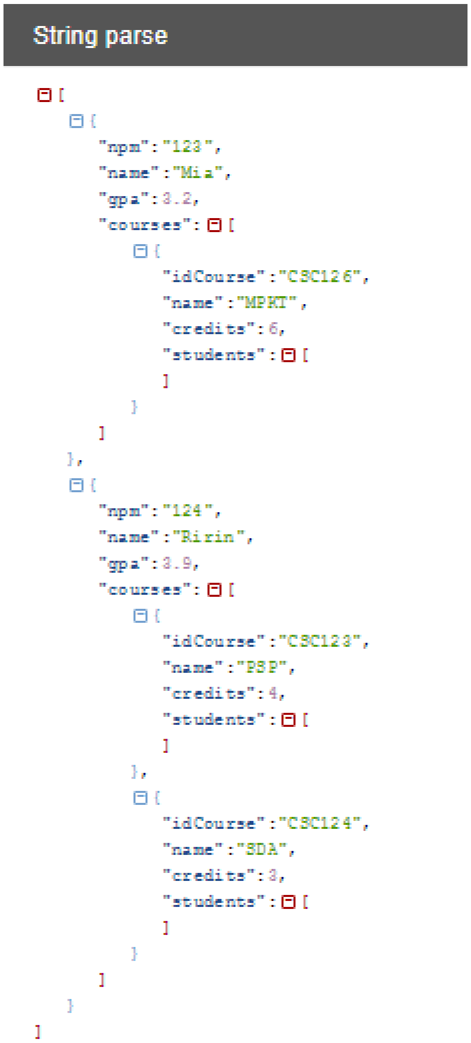
```
@RequestMapping("/student/viewall")
public List<StudentModel> viewall(StudentModel model) {
    List<StudentModel> students = studentService.selectAllStudents();
    return students;
}
```

Pemanggilan service viewall Student akan diakses melalui /rest/student/viewall. Untuk menyimpan daftar data semua Student pada variabel students maka digunakan List<StudentModel> dengan memanggil method selectAllStudents() pada service.

Mengakses **localhost:8080/rest/student/viewall**



```
[{"npm":"123","name":"Mia","gpa":3.2,"courses":
[{"idCourse":"CSC126","name":"MPKT","credits":6,"students":[]}]},
{"npm":"124","name":"Ririn","gpa":3.9,"courses":
[{"idCourse":"CSC123","name":"PSP","credits":4,"students":[]},
{"idCourse":"CSC124","name":"SDA","credits":3,"students":[]}]}]
```



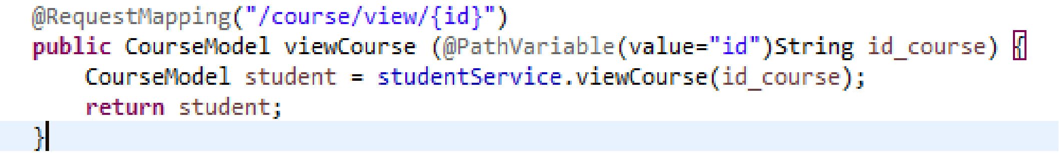
String parse

```
[
   {
      "npm":"123",
      "name":"Mia",
      "gpa":3.2,
      "courses":[
         {
            "idCourse":"CSC126",
            "name":"MPKT",
            "credits":6,
            "students":[
            ]
         }
      ]
   },
   {
      "npm":"124",
      "name":"Ririn",
      "gpa":3.9,
      "courses":[
         {
            "idCourse":"CSC123",
            "name":"PSP",
            "credits":4,
            "students":[
            ]
         },
         {
            "idCourse":"CSC124",
            "name":"SDA",
            "credits":3,
            "students":[
            ]
         }
      ]
   }
]
```

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).

a. view course by id

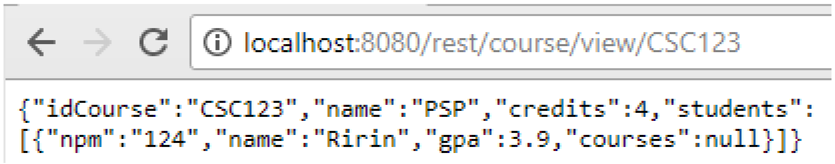
StudentRestController.java



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

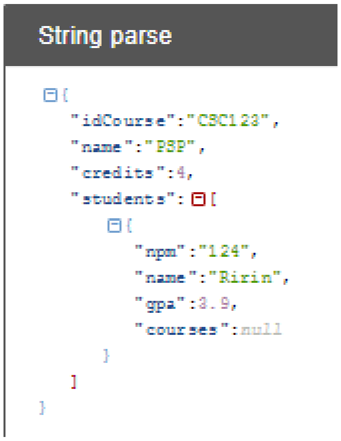
Pemanggilan service view Course by id_course akan diakses melalui /rest/course/view/{id}. Untuk menyimpan data course yang dipilih pada variabel student maka akan dipanggil method viewCourse(id_course) pada service.

Mengakses **localhost:8080/rest/course/view/CSC123**



localhost:8080/rest/course/view/CSC123

```
{"idCourse":"CSC123","name":"PSP","credits":4,"students":
[{"npm":"124","name":"Ririn","gpa":3.9,"courses":null}]}
```

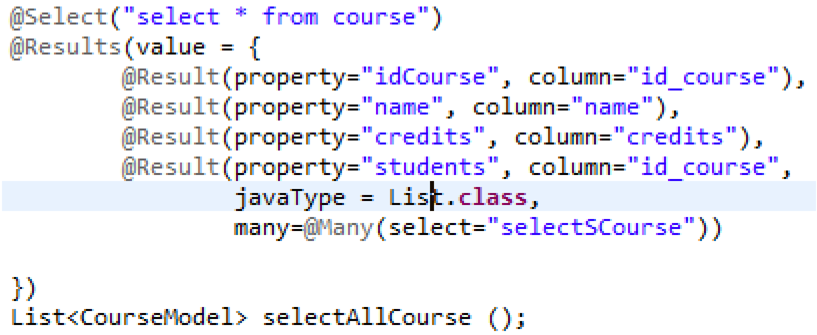


String parse

```
⊟{
   "idCourse":"CSC123",
   "name":"PSP",
   "credits":4,
   "students":⊟[
      ⊟{
         "npm":"124",
         "name":"Ririn",
         "gpa":3.9,
         "courses":null
      }
   ]
}
```

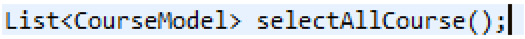
Untuk view all course, maka perlu ditambahkan terlebih dahulu method sebagai berikut :

StudentMapper.java



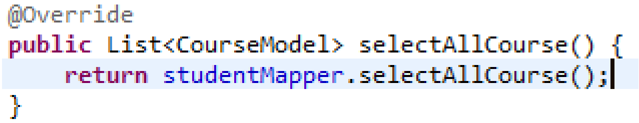
```
@Select("select * from course")
@Results(value = {
        @Result(property="idCourse", column="id_course"),
        @Result(property="name", column="name"),
        @Result(property="credits", column="credits"),
        @Result(property="students", column="id_course",
                javaType = List.class,
                many=@Many(select="selectSCourse"))

})
List<CourseModel> selectAllCourse ();
```

StudentService.java



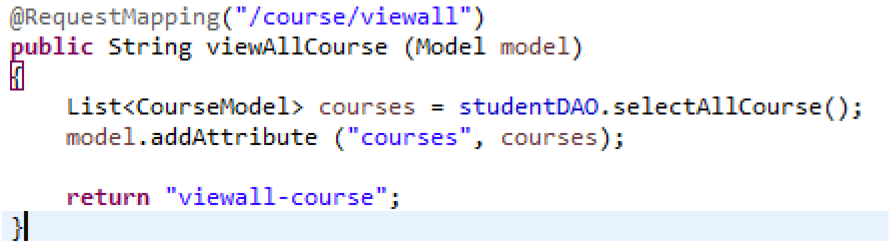
```
List<CourseModel> selectAllCourse();
```

StudentServiceDatabase.java



```
@Override
public List<CourseModel> selectAllCourse() {
    return studentMapper.selectAllCourse();
}
```

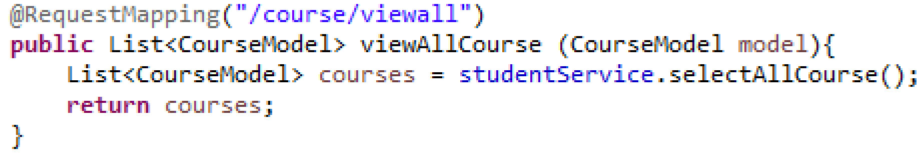
StudentController.java



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

    return "viewall-course";
}
```

b. viewall course

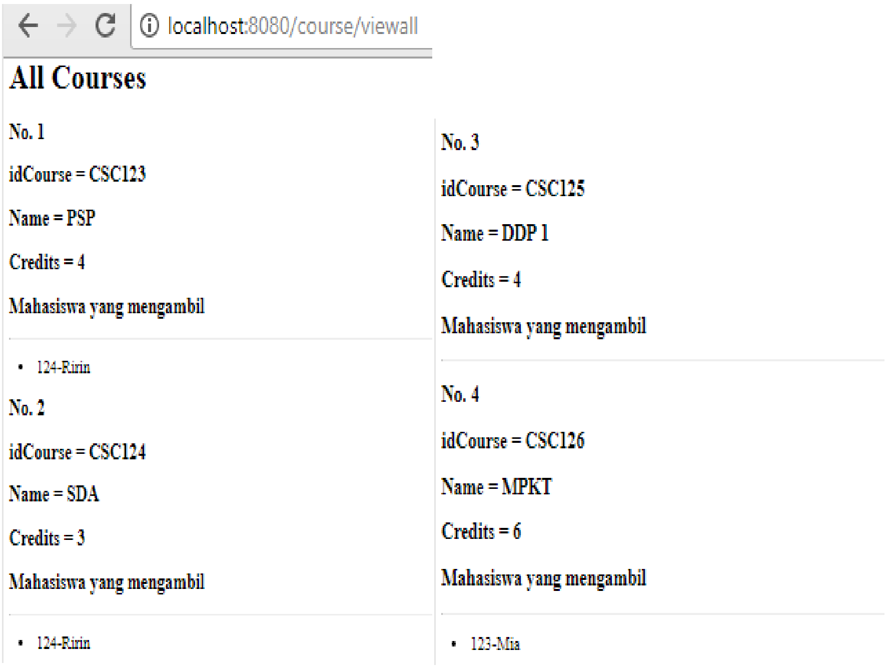
StudentRestController.java



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

Pemanggilan service viewall Course akan diakses melalui /rest/course/viewall. Untuk menyimpan daftar data semua Course pada variabel courses maka digunakan List<CourseModel> dengan memanggil method selectAllCourse() pada service.

Mengakses **localhots:8080/course/viewall**



localhost:8080/course/viewall

All Courses

No. 1

idCourse = CSC123

Name = PSP

Credits = 4

Mahasiswa yang mengambil

- 124-Ririn

No. 2

idCourse = CSC124

Name = SDA

Credits = 3

Mahasiswa yang mengambil

- 124-Ririn

No. 3

idCourse = CSC125

Name = DDP 1

Credits = 4

Mahasiswa yang mengambil

No. 4

idCourse = CSC126

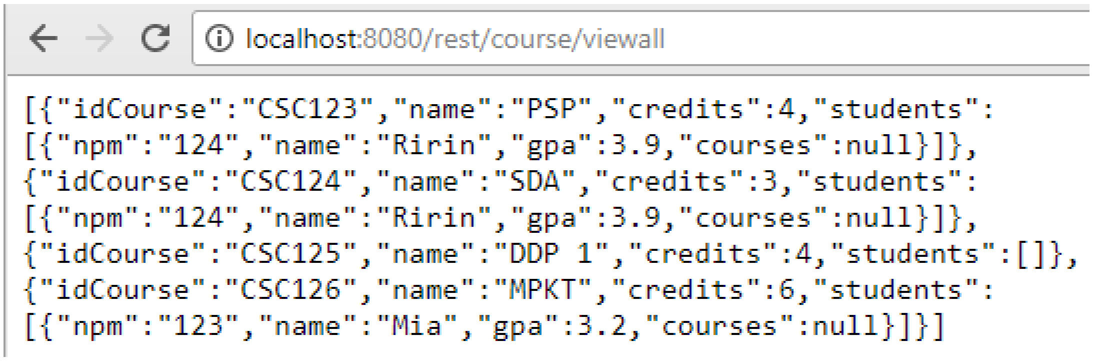
Name = MPKT

Credits = 6

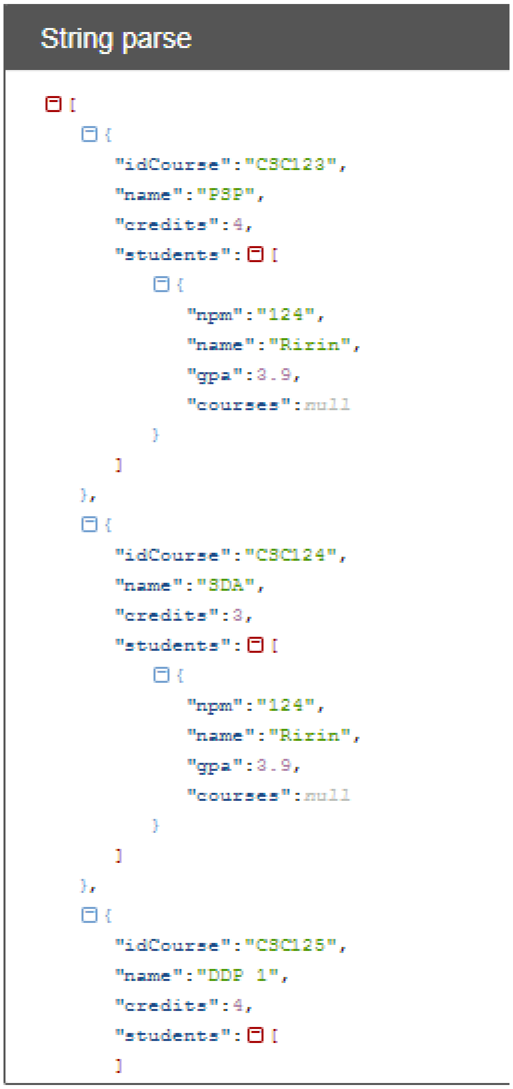
Mahasiswa yang mengambil

- 123-Mia

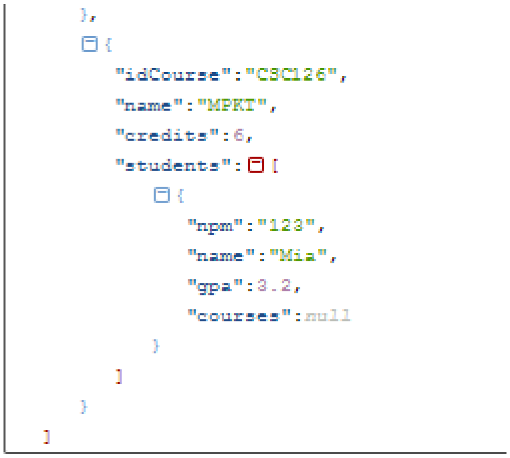
Mengakses **localhost:8080/rest/course/viewall**



```
[{"idCourse":"CSC123","name":"PSP","credits":4,"students":
[{"npm":"124","name":"Ririn","gpa":3.9,"courses":null}]},
{"idCourse":"CSC124","name":"SDA","credits":3,"students":
[{"npm":"124","name":"Ririn","gpa":3.9,"courses":null}]},
{"idCourse":"CSC125","name":"DDP 1","credits":4,"students":[]},
{"idCourse":"CSC126","name":"MPKT","credits":6,"students":
[{"npm":"123","name":"Mia","gpa":3.2,"courses":null}]}]
```



```
⊟ [
    ⊟ {
        "idCourse":"CSC123",
        "name":"PSP",
        "credits":4,
        "students":⊟ [
            ⊟ {
                "npm":"124",
                "name":"Ririn",
                "gpa":3.9,
                "courses":null
            }
        ]
    },
    ⊟ {
        "idCourse":"CSC124",
        "name":"SDA",
        "credits":3,
        "students":⊟ [
            ⊟ {
                "npm":"124",
                "name":"Ririn",
                "gpa":3.9,
                "courses":null
            }
        ]
    },
    ⊟ {
        "idCourse":"CSC125",
        "name":"DDP 1",
        "credits":4,
        "students":⊟ [
        ]
```

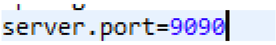


```
    },
    {
        "idCourse":"CSC126",
        "name":"MPKT",
        "credits":6,
        "students":[
            {
                "npm":"123",
                "name":"Mia",
                "gpa":3.2,
                "courses":null
            }
        ]
    }
]
```

Membuat Service Consumer

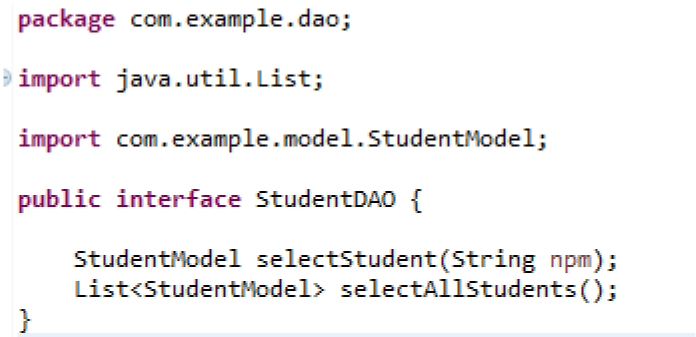
Service Consumer akan digunakan untuk mengkonsumsi web service sehingga dapat mengakses data-data Student dari Tutorial07Producer.

1. Mengimpor berkas-berkas yang ada pada Tutorial 6 untuk ServiceConsumer
2. Menambah baris server.port=9090 pada application properties



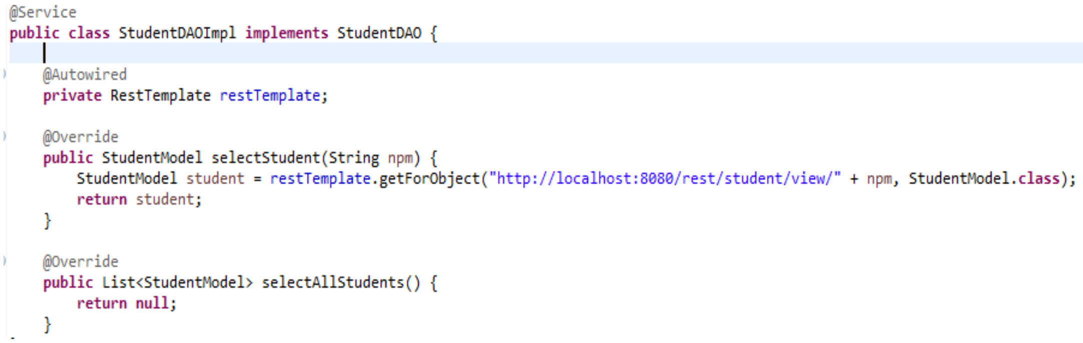
```
server.port=9090
```

3. Menambahkan kelas interface StudentDAO



```
package com.example.dao;

import java.util.List;

import com.example.model.StudentModel;

public interface StudentDAO {

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

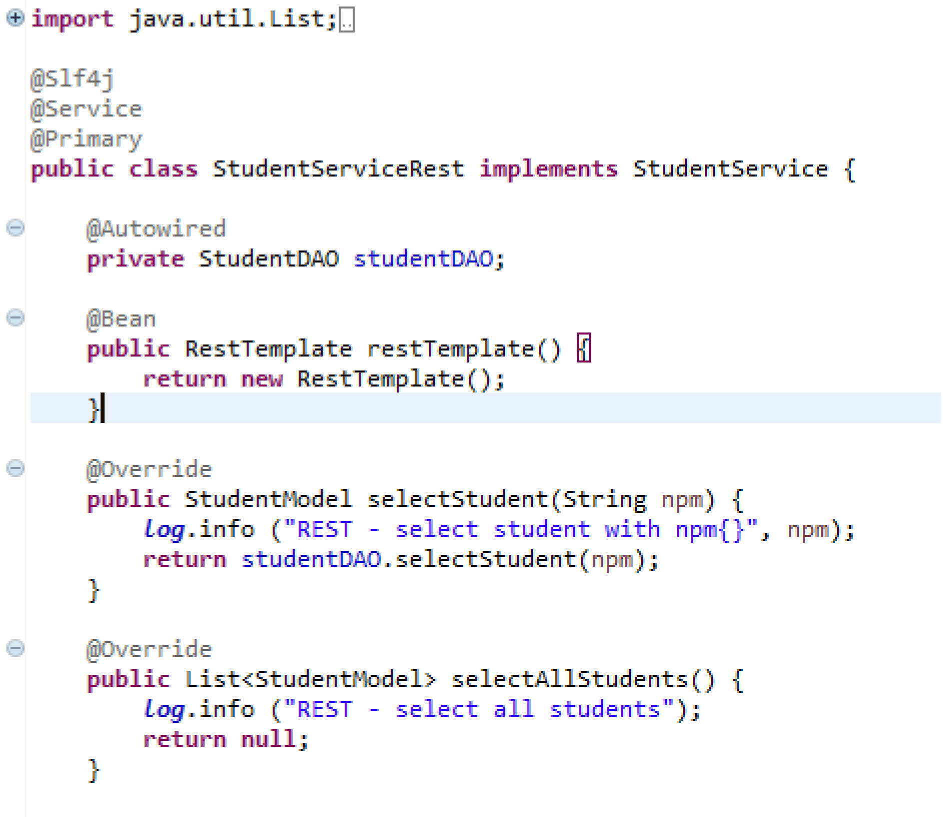
4. Menambahkan kelas StudentDAOImpl yang akan mengimplementasi kelas StudentDAO



```
@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;
    }

    @Override
    public List<StudentModel> selectAllStudents() {
        return null;
    }
```

getForObject digunakan untuk menerima parameter berupa URL producer web service dan tipe class object yang didapatkan.

5. Menambahkan kelas StudentServiceRest pada service

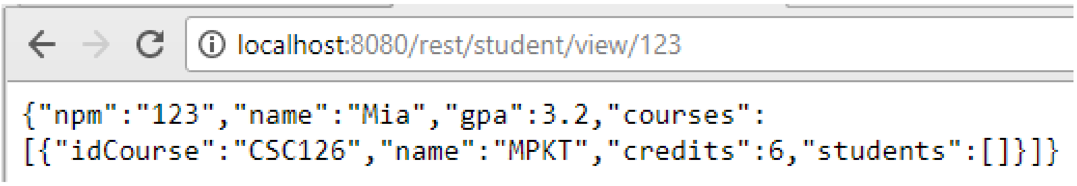


```
import java.util.List;

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

    @Autowired
    private StudentDAO studentDAO;

    @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 null;
    }
```

@Primary akan digunakan agar pada StudentController dapat mengambil data dari web service.

6. Menjalankan kedua project Service Producer dan Service Consumer
7. Mengakses **localhost:8080/rest/student/view/123**

   Hasilnya sebagai berikut :

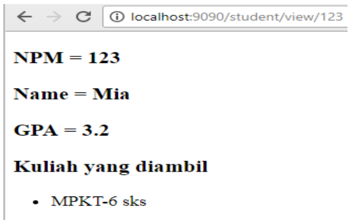


localhost:8080/rest/student/view/123

```
{"npm":"123","name":"Mia","gpa":3.2,"courses":
[{"idCourse":"CSC126","name":"MPKT","credits":6,"students":[]}]}
```

8. Menguji service consumer dengan mengakses **localhost:9090/student/view/123**

   Hasilnya sebagai berikut :



localhost:9090/student/view/123

NPM = 123

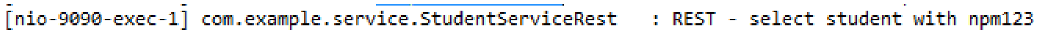
Name = Mia

GPA = 3.2

Kuliah yang diambil

- MPKT-6 sks

Pada console akan tertuli sebagai berikut :

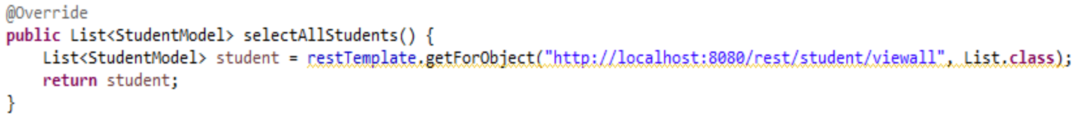


```
[nio-9090-exec-1] com.example.service.StudentServiceRest   : REST - select student with npm123
```

LATIHAN

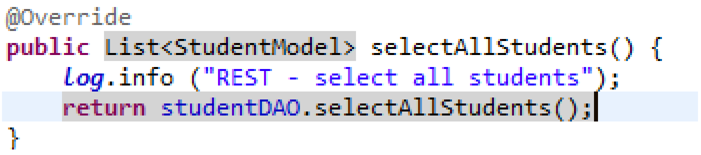
1. Implementasikan service consumer untuk view all Students dengan melengkapi method selectAllStudents yang ada di kelas StudentServiceRest. Cantumkan dan jelaskan method Anda pada write-up

StudentDAOImp.java



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

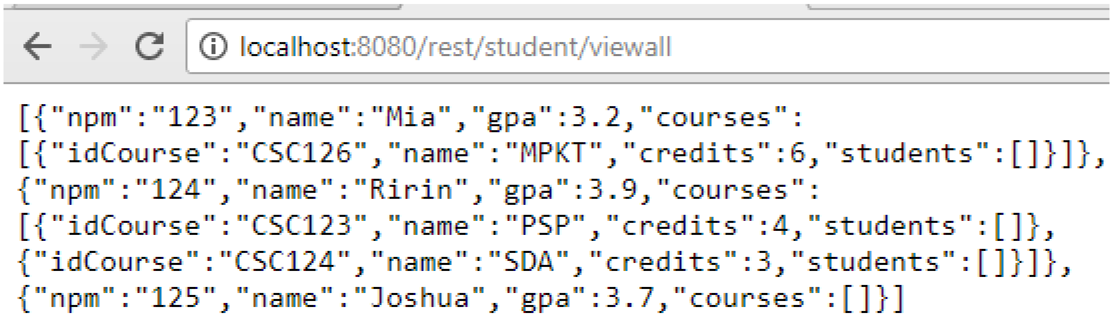
StudentServiceRest.java



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

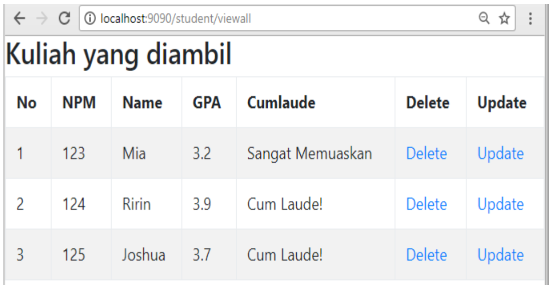
Digunakan kelas RestTemplate untuk meng-consume object REST web service. Method yang digunakan adalah getForObject yang menerima parameter berupa URL producer web service yaitu localhost:8080/rest/student/viewall dan tipe class dari object yang didapatkan yaitu berupa List.class. Lalu pada StudentServiceRest.java method List<StudentModel> selectAllStudents() akan mengembalikan hasil dari pemanggilan method selectAllStudents() pada studentDAO.

Mengakses **localhost:8080/rest/student/viewall**
Hasilnya sebagai berikut :



```
[{"npm":"123","name":"Mia","gpa":3.2,"courses":
[{"idCourse":"CSC126","name":"MPKT","credits":6,"students":[]}]},
{"npm":"124","name":"Ririn","gpa":3.9,"courses":
[{"idCourse":"CSC123","name":"PSP","credits":4,"students":[]},
{"idCourse":"CSC124","name":"SDA","credits":3,"students":[]}]},
{"npm":"125","name":"Joshua","gpa":3.7,"courses":[]}]
```

Mengakses **localhost:9090/student/viewall**



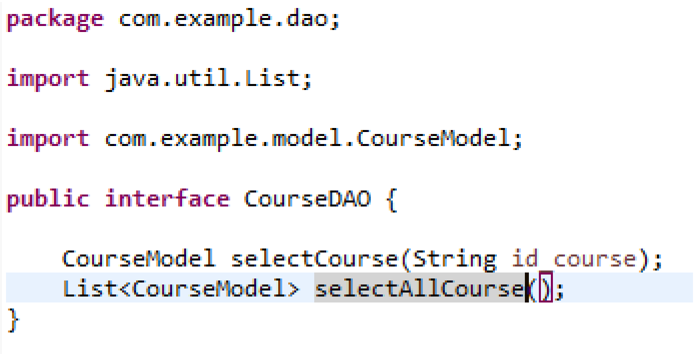
localhost:9090/student/viewall

Kuliah yang diambil

No	NPM	Name	GPA	Cumlaude	Delete	Update
1	123	Mia	3.2	Sangat Memuaskan	Delete	Update
2	124	Ririn	3.9	Cum Laude!	Delete	Update
3	125	Joshua	3.7	Cum Laude!	Delete	Update

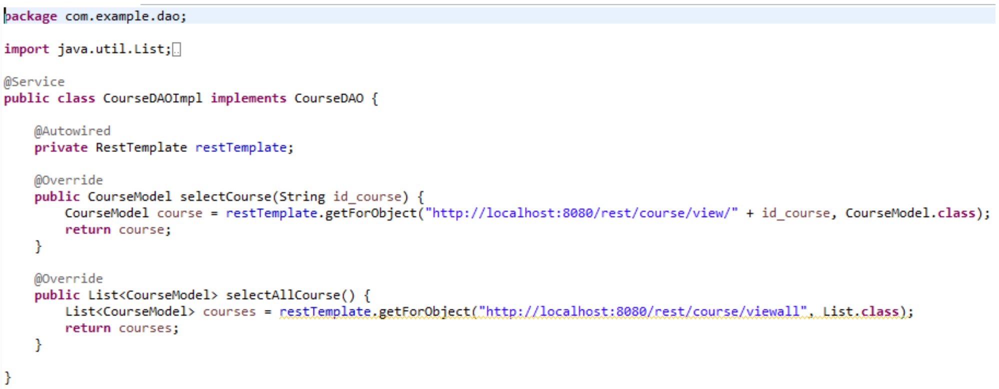
2. Implementasikan service consumer untuk class CourseModel dengan membuat class-class DAO dan service baru.

   CourseDAO.java



```java
package com.example.dao;

import java.util.List;

import com.example.model.CourseModel;

public interface CourseDAO {

    CourseModel selectCourse(String id_course);
    List<CourseModel> selectAllCourse();
}
```

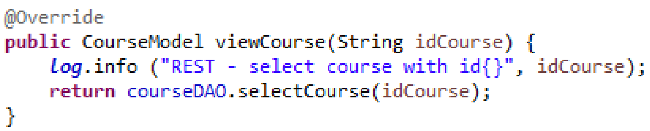
CourseDAOImpl.java



```java
package com.example.dao;

import java.util.List;

@Service
public class CourseDAOImpl implements CourseDAO {

    @Autowired
    private RestTemplate 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;
    }

}
```

a. view course by id

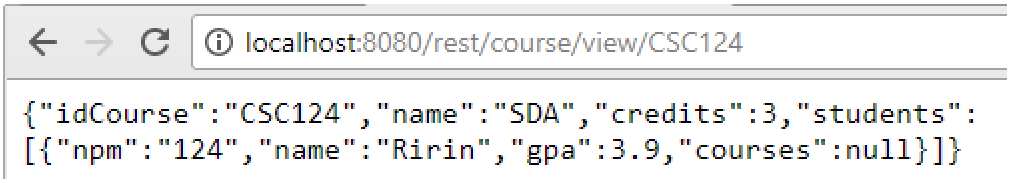
StudentServiceRest.java



```java
@Override
public CourseModel viewCourse(String idCourse) {
    log.info ("REST - select course with id{}", idCourse);
    return courseDAO.selectCourse(idCourse);
}
```

Digunakan kelas RestTemplate untuk meng-consume object REST web service. Method yang digunakan adalah getForObject yang menerima parameter berupa URL producer web service yaitu localhost:8080/rest/student/view/ + id_course dan tipe class dari object yang didapatkan yaitu berupa CourseModel.class. Lalu pada StudentServiceRest.java method viewCourse(String id_course) akan mengembalikan hasil dari pemanggilan method selectCourse(idCourse) pada courseDAO.

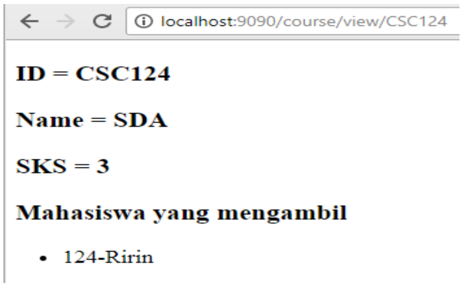
Mengakses **localhost:8080/rest/course/view/CSC124**
Hasilnya sebagai berikut :



localhost:8080/rest/course/view/CSC124

```
{"idCourse":"CSC124","name":"SDA","credits":3,"students":
[{"npm":"124","name":"Ririn","gpa":3.9,"courses":null}]}
```

Mengakses **localhost:9090/student/view/CSC124**



localhost:9090/course/view/CSC124

ID = CSC124

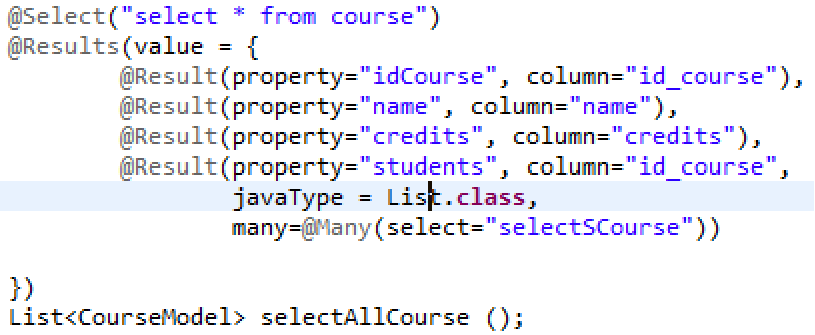
Name = SDA

SKS = 3

Mahasiswa yang mengambil

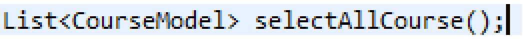
- 124-Ririn

Untuk view all course, maka perlu ditambahkan terlebih dahulu method sebagai berikut :
StudentMapper.java



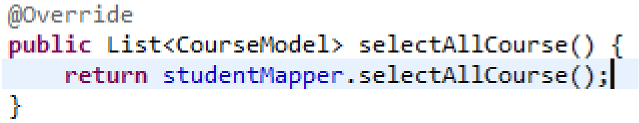
```
@Select("select * from course")
@Results(value = {
        @Result(property="idCourse", column="id_course"),
        @Result(property="name", column="name"),
        @Result(property="credits", column="credits"),
        @Result(property="students", column="id_course",
                javaType = List.class,
                many=@Many(select="selectSCourse"))

})
List<CourseModel> selectAllCourse ();
```

StudentService.java



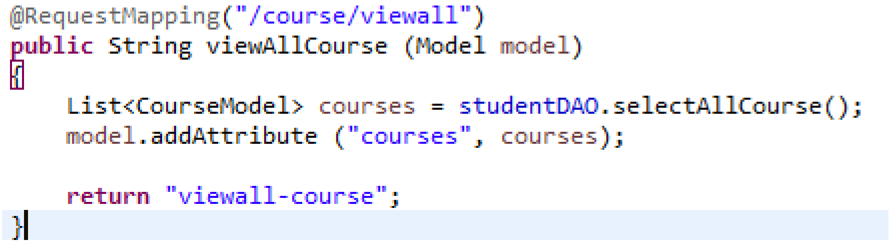
```
List<CourseModel> selectAllCourse();
```

StudentServiceDatabase.java



```
@Override
public List<CourseModel> selectAllCourse() {
    return studentMapper.selectAllCourse();
}
```

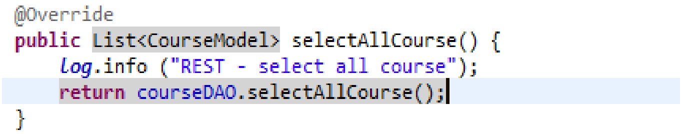
StudentController.java



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

    return "viewall-course";
}
```

b. viewall course

StudentServiceRest.java

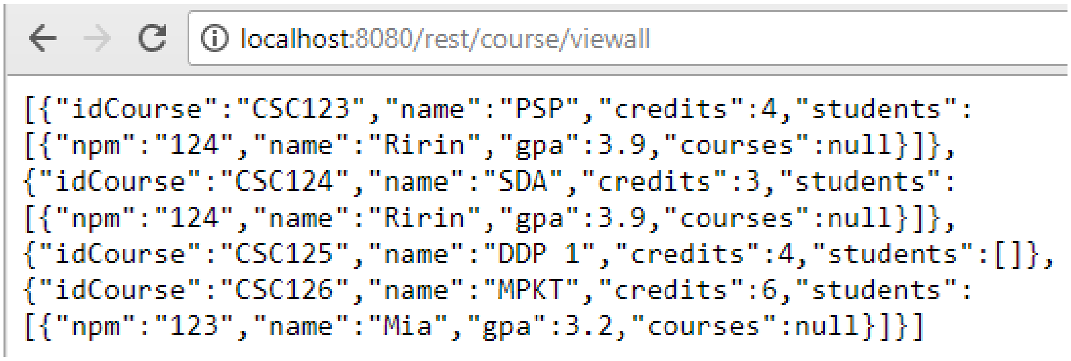


```
@Override
public List<CourseModel> selectAllCourse() {
    log.info ("REST - select all course");
    return courseDAO.selectAllCourse();
}
```

Digunakan kelas RestTemplate untuk meng-consume object REST web service. Method yang digunakan adalah getForObject yang menerima parameter berupa URL producer web service yaitu localhost:8080/rest/course/viewall dan tipe class dari object yang didapatkan yaitu berupa List.class. Lalu pada StudentServiceRest.java method List<CourseModel> selectAllCourse() akan mengembalikan hasil dari pemanggilan method selectAllCourse() pada CourseDAO.

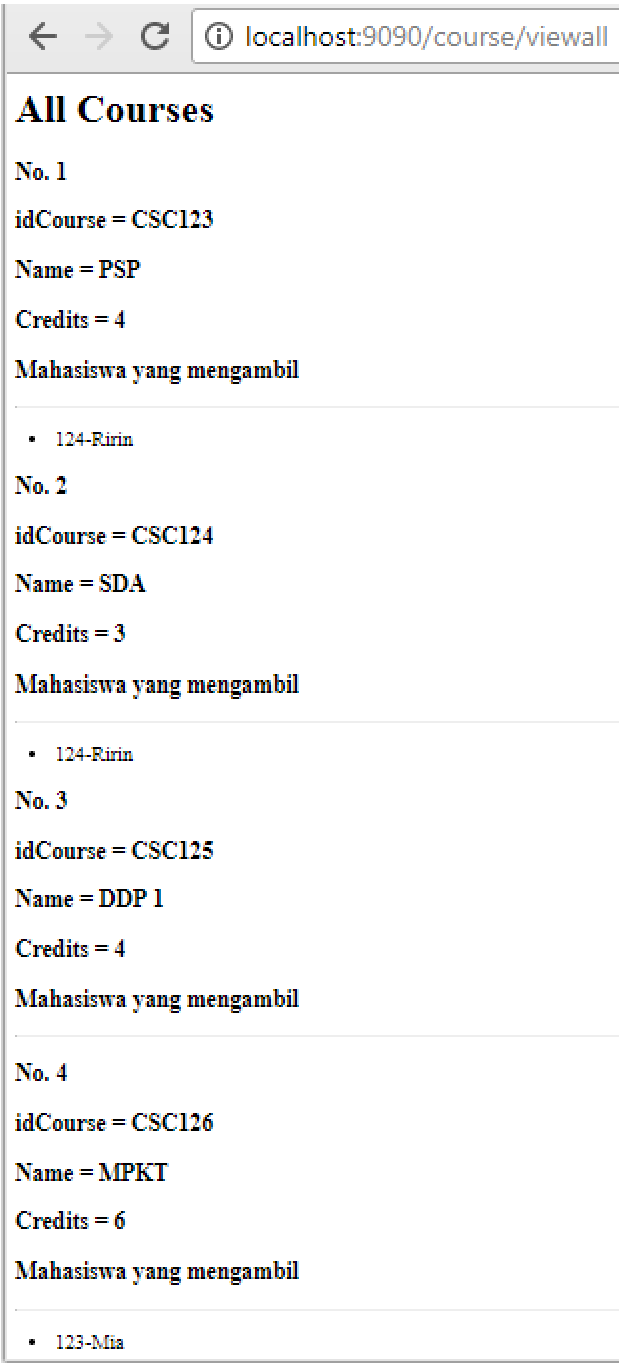
Mengakses **localhost:8080/rest/course/viewall**

Hasilnya sebagai berikut :



```
[{"idCourse":"CSC123","name":"PSP","credits":4,"students":
[{"npm":"124","name":"Ririn","gpa":3.9,"courses":null}]},
{"idCourse":"CSC124","name":"SDA","credits":3,"students":
[{"npm":"124","name":"Ririn","gpa":3.9,"courses":null}]},
{"idCourse":"CSC125","name":"DDP 1","credits":4,"students":[]},
{"idCourse":"CSC126","name":"MPKT","credits":6,"students":
[{"npm":"123","name":"Mia","gpa":3.2,"courses":null}]}]
```

Mengakses **localhost:9090/student/viewall**



All Courses

No. 1

idCourse = CSC123

Name = PSP

Credits = 4

Mahasiswa yang mengambil

- 124-Ririn

No. 2

idCourse = CSC124

Name = SDA

Credits = 3

Mahasiswa yang mengambil

- 124-Ririn

No. 3

idCourse = CSC125

Name = DDP 1

Credits = 4

Mahasiswa yang mengambil

No. 4

idCourse = CSC126

Name = MPKT

Credits = 6

Mahasiswa yang mengambil

- 123-Mia