

## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B

# MEMBUAT WEB SERVICE MENGGUNAKAN SPRING BOOT FRAMEWORK

---

## A. HAL YANG DIPELAJARI

Dalam pengerjaan tutorial ketujuh mata kuliah Arsitektur dan Pemrograman Aplikasi Perusahaan, saya mempelajari mengenai pembuatan *web service* menggunakan Spring Boot *Framework*. Disini saya memahami bahwa pada kenyataannya, membangun sebuah sistem informasi berskala besar tidak mungkin hanya mempunyai sebuah aplikasi saja. Sistem informasi tersebut mempunyai aplikasi yang tersebar pada beberapa mesin *server*.

Di tutorial ini, memahami tentang pemisahan *layer backend (service producer)* dan *layer frontend (service consumer)*. *Service producer* menyediakan data dari *database* yang biasanya tidak memiliki *view* untuk dapat dilihat pengguna, *service producer* akan memberikan data untuk *service consumer*. Sementara *service consumer* adalah aplikasi yang berinteraksi dengan pengguna dengan menyediakan dan mengolah data untuk pengguna tanpa perlu memiliki *database*.

Dalam hal ini, komunikasi antara *service consumer* dengan *service producer*, dijembatani oleh *web service* yang disediakan *service producer* untuk dikonsumsi oleh *service consumer*. *Web service* adalah sebuah URL yang mengembalikan data dalam representasi yang telah disepakati, di tutorial ini representasi data yang digunakan adalah dalam bentuk JSON.

Tutorial ini membantu saya dalam mempelajari pembuatan *web service* untuk mengakses data-data Student untuk *service producer* dan *service consumer*. Nantinya JSON yang dihasilkan oleh *service producer* akan diolah oleh *service consumer* untuk ditampilkan pada halaman *view*. Jadi dalam tutorial ini terdapat adanya pemisahan tanggung jawab antara *service producer* yang mengambil data dari *database* dan *service consumer* yang mengambil data dari *object* yang dikembalikan *service producer* dalam representasi JSON.

## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B

## B. MENGERJAKAN LATIHAN

### 1. LATIHAN 1

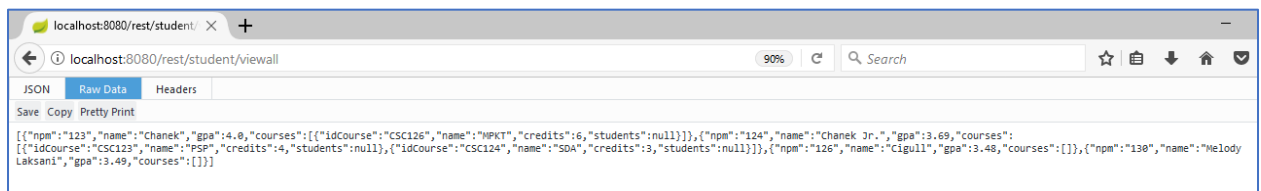
Buatlah service untuk mengembalikan seluruh student yang ada di basis data. Service ini mirip seperti method `viewAll` di Web Controller. Service tersebut di-mapping ke `"/rest/student/viewall"`.

- Pada `StudentRestController.java` membuat *method* untuk melakukan *viewall* Student yang mengembalikan *list of object* `StudentModel`.

```
/*Latihan 1*/
@RequestMapping("/student/viewall")
public List<StudentModel> viewall(){
    List<StudentModel> students = studentService.selectAllStudents();
    return students;
}
```

*Method* ini akan dijalankan ketika pada *browser* diketikkan `http:8080/rest/student/viewall`. *Method* ini membentuk *list of object* `StudentModel` dengan memanggil method `selectAllStudents()` pada *interface* `StudentService` yang diimplementasikan pada *class* `StudentServiceDatabase` yang mengembalikan kumpulan dari `StudentModel` sesuai dengan data Student yang ada pada *database*. Nantinya *list of object* `StudentModel` ini akan dikembalikan oleh *method* `viewall()` pada *class* `StudentRestController`.

- Ketika program dijalankan dan mengakses `http:8080/rest/student/viewall` pada *browser*, akan menghasilkan *output* seperti berikut :



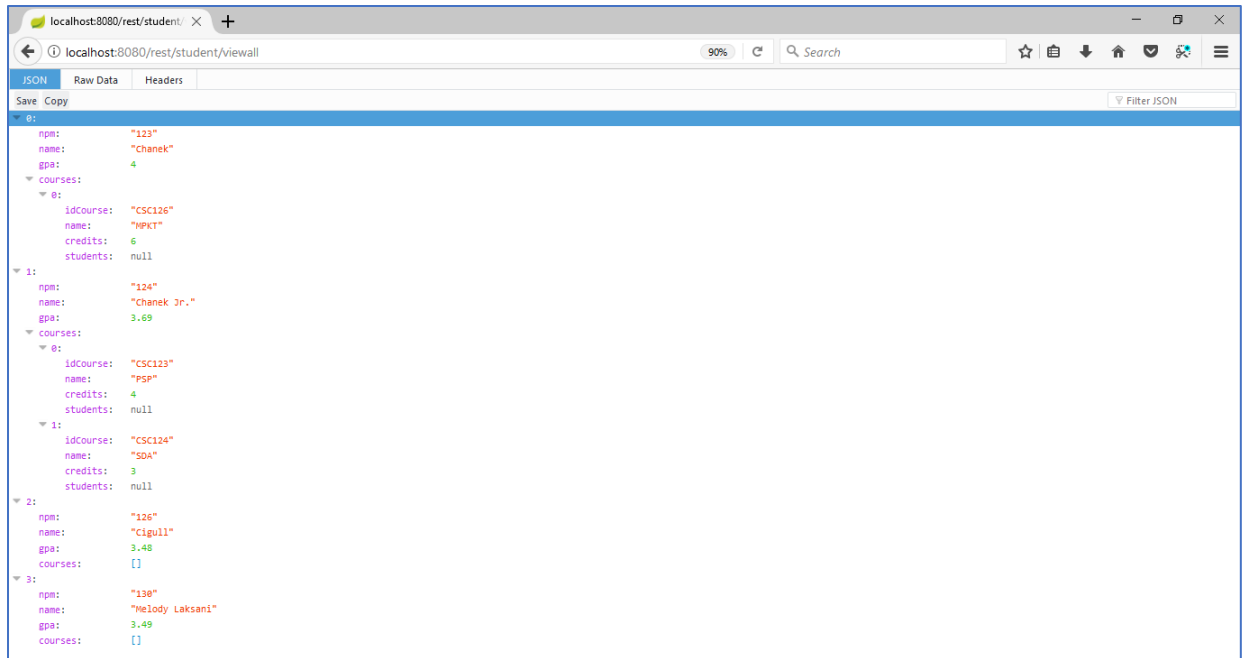
## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B

Dalam bentuk JSON view



## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B

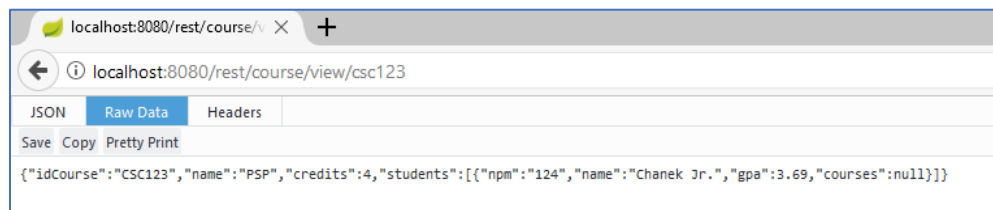
### 2. LATIHAN 2

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*).

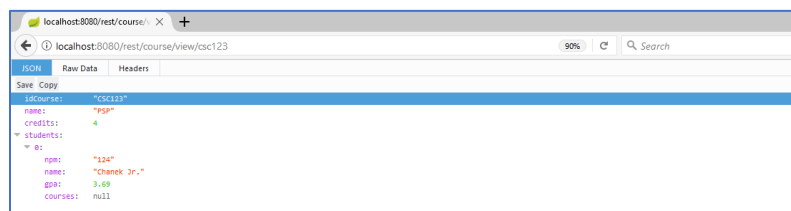
- Membuat *class* CourseRestController.java

```
13 @RestController
14 @RequestMapping("/rest")
15 public class CourseRestController {
16     @Autowired
17     StudentService courseService;
18
19     /*Latihan 2*/
20     @RequestMapping("/course/view/{id}")
21     public CourseModel courseView (@PathVariable(value = "id") String id) {
22         CourseModel course = courseService.selectCourse(id);
23         return course;
24     }
25
26     @RequestMapping("/course/viewall")
27     public List<CourseModel> courseViewAll(){
28         List<CourseModel> courses = courseService.selectAllCourses();
29         return courses;
30     }
31 }
32
33 }
```

- *Method* courseView untuk melihat atau menampilkan Course sesuai id yang dimasukkan. Id yang di-input dijadikan parameter untuk memanggil *method* selectCourse pada *interface* StudentService. *Method* ini akan mengembalikan *object* CourseModel. Ketika menjalankan aplikasi dan pada *browser* mengakses **localhost:8080/rest/course/view/csc123** , maka akan muncul tampilan seperti berikut :



Dalam bentuk JSON view :



Ini adalah contoh ketika kita ingin menampilkan Course dengan id CSC123.

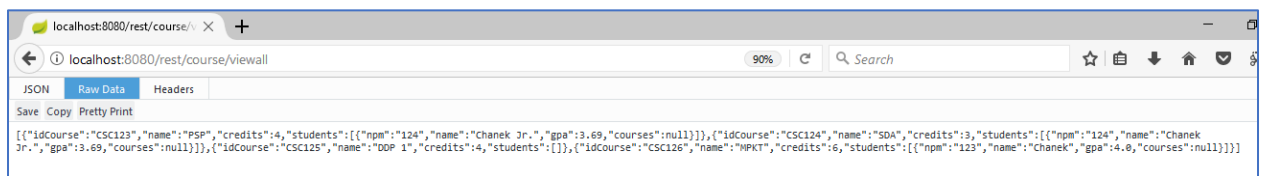
## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

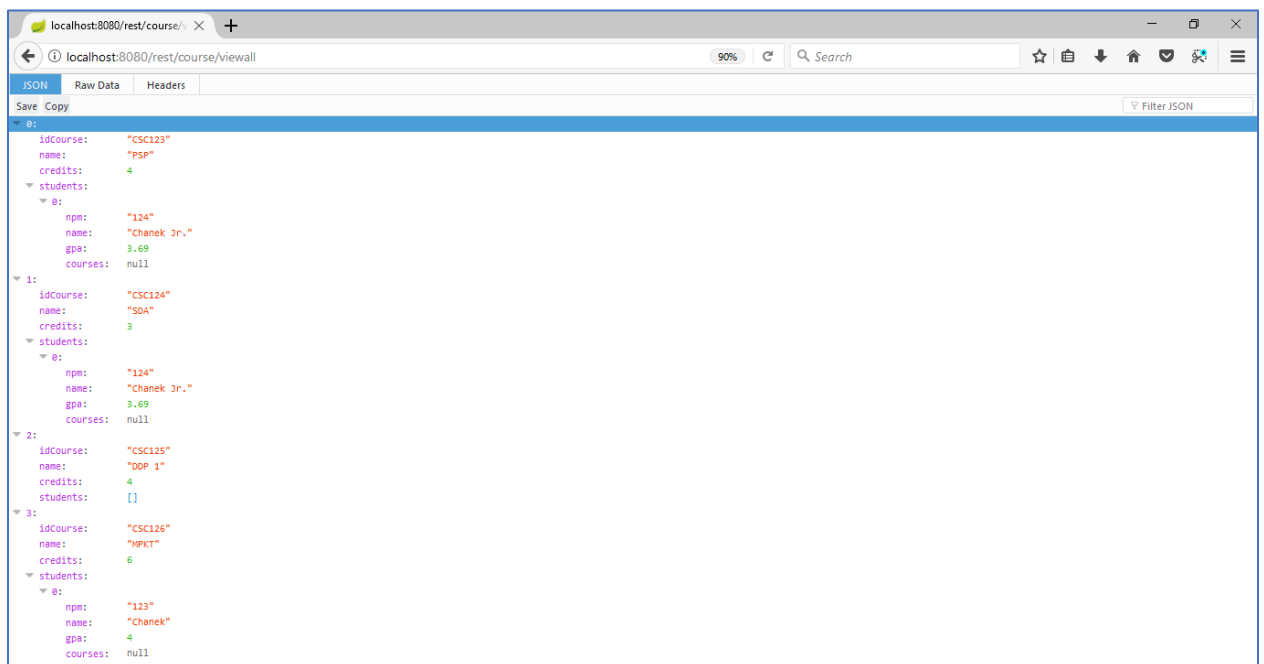
NPM : 1506689622

Kelas : ADPAP – B

- *Method* `courseViewAll` untuk menampilkan seluruh *Course* yang ada pada *database* dalam bentuk *list of object* *CourseModel*. *Method* ini membentuk *list of object* *CourseModel* dengan memanggil *method* `selectAllCourses()` pada *interface* *StudentService* yang diimplementasikan pada *class* *StudentServiceDatabase* yang mengembalikan kumpulan dari *CourseModel* sesuai dengan data *Course* yang ada pada *database*. Nantinya *list of object* *CourseModel* ini akan dikembalikan oleh *method* `courseViewAll()` pada *class* *CourseRestController*. Ketika menjalankan aplikasi dan pada *browser* mengakses **localhost:8080/rest/course/viewall** , maka akan muncul tampilan seperti berikut :



Dalam bentuk *JSON view*



## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B

### 3. LATIHAN 3

Implementasikan service consumer untuk view all Students dengan melengkapi method `selectAllStudents` yang ada di kelas `StudentServiceRest`.

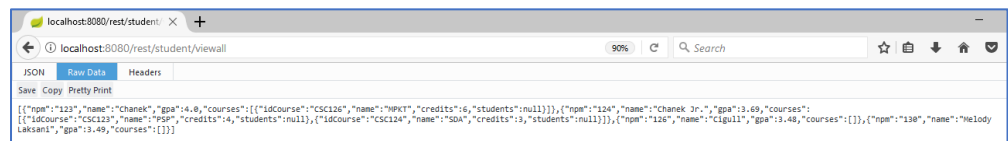
- Pada *class* `StudentDAOImpl`, membuat *method* yang mengembalikan *list of object* `StudentModel`. Mirip dengan `StudentMapper`, bedanya dalam *class* `StudentDAOImpl` mengambil data yang dikembalikan oleh *RestController* dari *service producer* ketika diakses `localhost:8080/rest/student/viewall`.

```
29 @Override
30 public List<StudentModel> selectAllStudents(){
31     List<StudentModel> students = restTemplate.getForObject("http://localhost:8080/rest/student/viewall", List.class);
32     return students;
33 }
```

- Melengkapi *method* `selectAllStudents` pada *class* `StudentServiceRest` dengan memanggil *method* `selectAllStudents` pada *interface* `StudentDAO` yang diimplementasikan oleh *class* `StudentDAOImpl`.

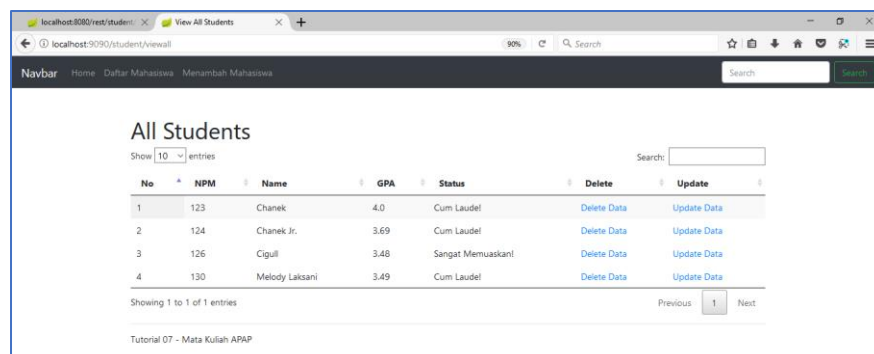
```
32 @Override
33 public List<StudentModel> selectAllStudents(){
34     Log.info("REST - select all students");
35     return studentDAO.selectAllStudents();
36 }
37
```

- Untuk menjalankan *service consumer*, kita perlu memastikan bahwa *service producer* sudah berjalan dengan menjalankan `localhost:8080/rest/student/viewall`



```
[{"npn": "123", "name": "Chanek", "gpa": 4.0, "courses": [{"idcourse": "CSC126", "name": "NPCT", "credits": 6, "students": null}], {"npn": "124", "name": "Chanek Jr.", "gpa": 3.69, "courses": [{"idcourse": "CSC123", "name": "PSP", "credits": 4, "students": null}, {"idcourse": "CSC124", "name": "SDA", "credits": 3, "students": null}], {"npn": "126", "name": "Cigull", "gpa": 3.48, "courses": []}, {"npn": "130", "name": "Melody Laksani", "gpa": 3.49, "courses": []}]
```

- Kemudian untuk menguji *service consumer* membuka `localhost:9090/student/viewall`



No	NPM	Name	GPA	Status	Delete	Update
1	123	Chanek	4.0	Cum Laude!	Delete Data	Update Data
2	124	Chanek Jr.	3.69	Cum Laude!	Delete Data	Update Data
3	126	Cigull	3.48	Sangat Memuaskan!	Delete Data	Update Data
4	130	Melody Laksani	3.49	Cum Laude!	Delete Data	Update Data

Showing 1 to 1 of 1 entries

Previous 1 Next

Tutorial 07 - Mata Kuliah APAP

## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B

- Pada *console* menampilkan :

```
2017-11-04 20:06:49.288 INFO 14472 --- [nio-9090-exec-1] com.example.service.StudentServiceRest : REST - select all students
```

Artinya *service consumer* menerima atau mengkonsumsi serta mengolah data dari *service producer* dan bukan mengambil langsung dari *database*.

Karena pada *StudentServiceRest* sudah menjadi *service primary* maka pengambilan data tidak lagi dilakukan oleh *StudentServiceDatabase*. Pada *controller class* *StudentController* menerima data yang akan ditampilkan pada halaman *view* melalui *method* pada *StudentServiceRest*

## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B

### 4. LATIHAN 4

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

- Membuat *interface* CourseDAO

```
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 idCourse);
9     List<CourseModel> selectAllCourses();
10 }
11
```

- Membuat *class* CourseDAOImpl yang mengimplementasikan *interface* CourseDAO serta melengkapi *method* selectCourse dan selectAllCourses. Di *class* ini akan meng-consume data dari apa yang dikembalikan di *RestController* ketika diakses **localhost:8080/rest/course/view/{idCourse}** atau **localhost:8080/rest/course/viewall**.

```
1 package com.example.dao;
2
3 import java.util.List;
4
11
12 @Service
13 public class CourseDAOImpl implements CourseDAO {
14     @Autowired
15     private RestTemplate restTemplate;
16
17     // method untuk handle error autowired untuk rest template
18     @Bean
19     public RestTemplate restTemplate() {
20         return new RestTemplate();
21     }
22
23
24     @Override
25     public CourseModel selectCourse(String idCourse) {
26         CourseModel course = restTemplate.getForObject("http://localhost:8080/rest/course/view/"+idCourse, CourseModel.class);
27         return course;
28     }
29
30     @Override
31     public List<CourseModel> selectAllCourses(){
32         List<CourseModel> courses = restTemplate.getForObject("http://localhost:8080/rest/course/viewall", List.class);
33         return courses;
34     }
35 }
36
```

*Method* selectCourse akan memilih Course berdasarkan id yang menjadi parameternya. *Method* ini akan mengembalikan CourseModel.

*Method* selectAllCourse akan memilih semua Course yang ada. *Method* ini akan mengembalikan *list of object* CourseModel (List<CourseModel>)



## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B

Nantinya kedua *method* ini akan dimanfaatkan pada *class service* untuk REST.

- Membuat *interface* CourseService

```
1 package com.example.service;
2
3 import java.util.List;
4
5
6
7 public interface CourseService {
8     CourseModel selectCourse(String idCourse);
9     List<CourseModel> selectAllCourses();
10 }
```

- Membuat *class service* CourseServiceRest yang mengimplementasikan *interface* CourseService yang nantinya akan digunakan pada *controller class* StudentController.

```
1 package com.example.service;
2
3 import java.util.List;
4
5
6
7
8
9
10
11
12
13
14 @Slf4j
15 @Service
16 @Primary
17 public class CourseServiceRest implements CourseService{
18     @Autowired
19     private CourseDAO courseDAO;
20
21
22     @Override
23     public CourseModel selectCourse(String idCourse) {
24         log.info("REST - select course with idcourse {}", idCourse);
25         return courseDAO.selectCourse(idCourse);
26     }
27
28     @Override
29     public List<CourseModel> selectAllCourses(){
30         log.info("REST - select all courses");
31         return courseDAO.selectAllCourses();
32     }
33 }
```

*Method* selectCourse akan memanggil *method* selectCourse pada CourseDAO yang diimplementasikan di *class* CourseDAOImpl yang mengembalikan *object* CourseModel.

*Method* selectAllCourses akan memanggil *method* selectAllCourses pada CourseDAO yang diimplementasikan di *class* CourseDAOImpl yang mengembalikan *list of object* CourseModel.

## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B

- Menambahkan *attribute object* CourseService yang diimplementasikan *method*-nya oleh CourseServiceRest dengan nama *variable* courseDAO pada *class* StudentController.

```
@Controller
public class StudentController
{
    @Autowired
    StudentService studentDAO;

    @Autowired
    CourseService courseDAO;
```

- Membuat *method* viewCourse dan viewallCourse pada *class* StudentController.

```
// Untuk Halaman view COURSE
@RequestMapping("/course/view/{idCourse}")
public String viewCourse(Model model,
    @PathVariable(value = "idCourse") String idCourse) {
    CourseModel course = courseDAO.selectCourse(idCourse);
    if (course != null) {
        model.addAttribute("course", course);
        return "view-course";
    } else {
        model.addAttribute("idCourse", idCourse);
        return "not-found-course";
    }
}

// Untuk Halaman view All COURSE
@RequestMapping("/course/viewall")
public String viewallCourse(Model model) {
    List<CourseModel> courses = courseDAO.selectAllCourses();
    model.addAttribute("courses", courses);
    return "viewall-course";
}
```

- Karena *view* yang tersedia pada Tutorial 6 hanya view-course.html, maka dibuat pula viewall-course.html sebagai halaman *view* yang menampilkan semua Course.

```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head th:replace="fragments/fragment :: head">
4 </head>
5 <body>
6 <script>
7     $(document).ready(function(){
8         $('#tableStudents').DataTable();
9     });
10 </script>
11 <div th:replace="fragments/fragment :: header"></div>
12 <br/>
13 <br/>
14 <div class="container">
15 <h1>All Courses</h1>
16 <table id="tableStudents" class="display">
17 <thead>
18 <th>No</th>
19 <th>ID Course</th>
20 <th>Course Name</th>
21 <th>Credits</th>
22 </thead>
23 <tbody th:each="course, iterationStatus: ${courses}" th:class="${iterationStatus.odd}? 'odd'">
24 <td th:text="${iterationStatus.count}">No</td>
25 <td th:text="${course.idCourse}">ID Course</td>
26 <td th:text="${course.name}">Name Course</td>
27 <td th:text="${course.credits}">Credits</td>
28 </tbody>
29 </table>
30 <div th:replace="fragments/fragment :: footer"></div>
31 </div>
32
33 </body>
```

## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

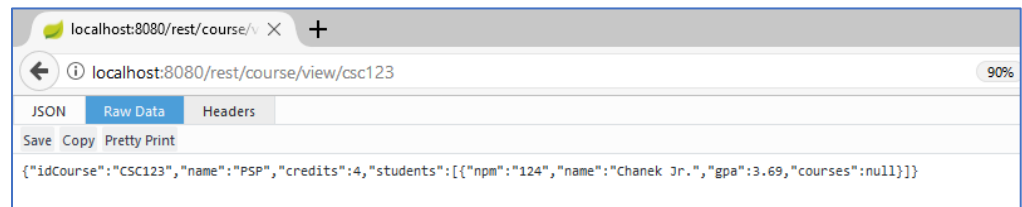
NPM : 1506689622

Kelas : ADPAP – B

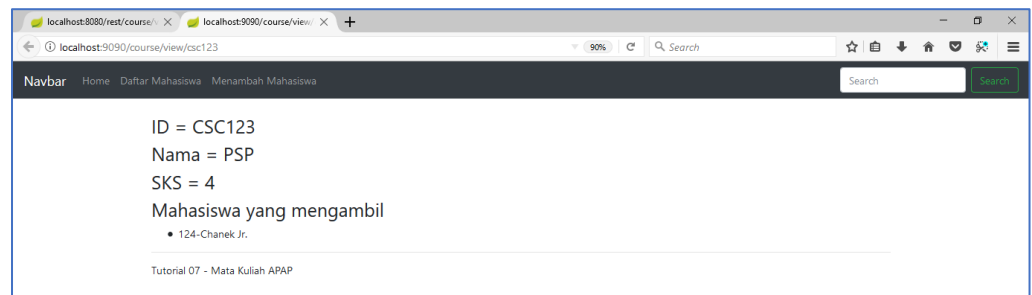
- Pengujian *service consumer* untuk *view course by id*

Misalnya ingin menampilkan *course* dengan id CSC123 :

Memastikan bahwa *service producer* sudah berjalan dengan menjalankan **localhost:8080/rest/course/view/csc123**



Mengakses **localhost:9090/course/view/csc123**

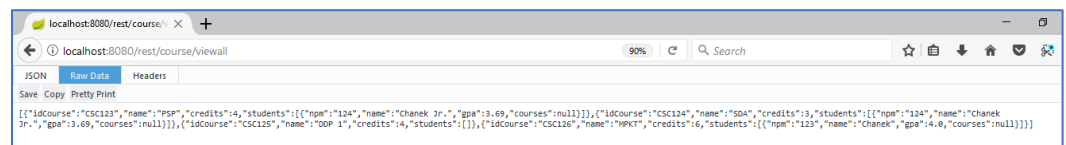


Pada *console* Tutorial07Consumer

```
[nio-9090-exec-7] com.example.service.CourseServiceRest : REST - select course with idcourse csc123
```

- Pengujian *service consumer* untuk *view all course*

Memastikan bahwa *service producer* sudah berjalan dengan menjalankan **localhost:8080/rest/course/viewall**



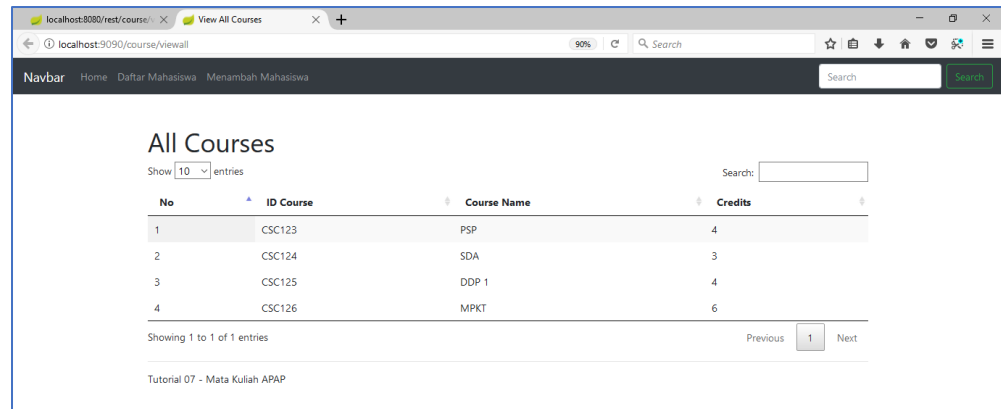
Mengakses **localhost:9090/course/viewall**

## WRITE-UP TUTORIAL 7 ADPAP

Nama : Yosua Bisma Putrapratama

NPM : 1506689622

Kelas : ADPAP – B



Pada *console* Tutorial07Consumer

```
[nio-9090-exec-2] com.example.service.CourseServiceRest : REST - select all courses
```