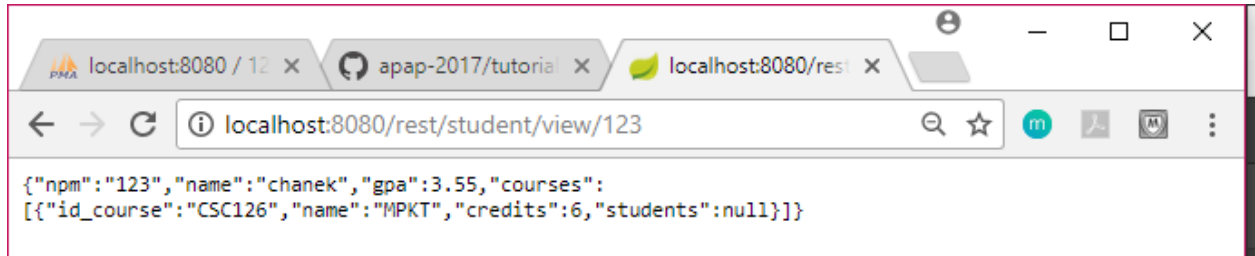


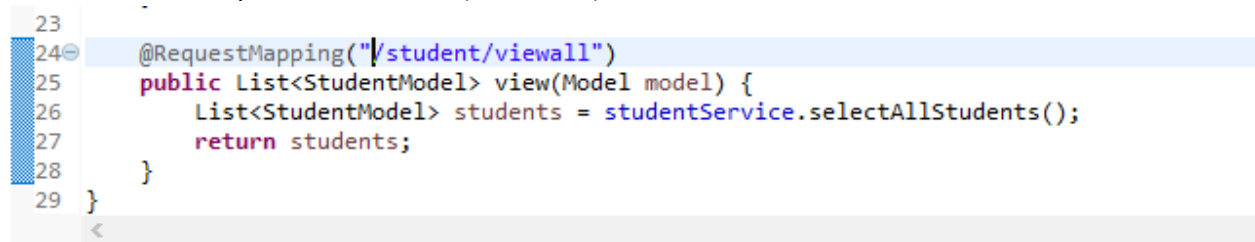
## Tutorial 07 Write-up

### 1. Tutorial07Producer

- Membuka halaman [localhost:8080/rest/student/view/123](http://localhost:8080/rest/student/view/123)



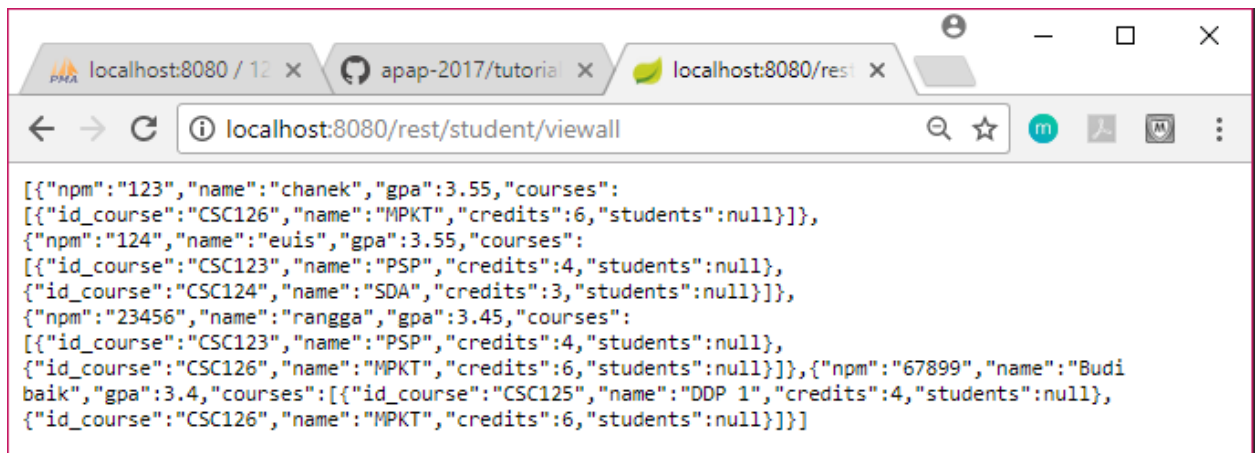
- Membuat *service producer* viewall (**Latihan 1**)



Penjelasan :

Pada *class* StudentRestController dibuat *method* view dengan RequestMapping method ini adalah `"/student/viewall"`, yang mengembalikan List yang berisi *students* yang ada di *database* dalam bentuk `List<StudentModel>`. *Method* ini akan dijalankan ketika halaman [localhost:8080/rest/student/viewall](http://localhost:8080/rest/student/viewall) di akses.

Hasil :



Atikah Luthfiana

1506689250

APAP B

- Membuat *service producer* untuk kelas Course, dan membuat *method view by id\_course* dan *viewall course* (**Latihan 2**)

Membuat class *CourseRestController*

```
1 package com.example.rest;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.web.bind.annotation.PathVariable;
5 import org.springframework.web.bind.annotation.RequestMapping;
6 import org.springframework.web.bind.annotation.RestController;
7 import com.example.model.CourseModel;
8 import com.example.service.CourseService;
9
10 @RestController
11 @RequestMapping("/rest")
12 public class CourseRestController {
13     @Autowired
14     CourseService courseService;
15
16 }
17
18 }
```

Membuat *method view course by id*

```
1 package com.example.rest;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.web.bind.annotation.PathVariable;
5 import org.springframework.web.bind.annotation.RequestMapping;
6 import org.springframework.web.bind.annotation.RestController;
7 import com.example.model.CourseModel;
8 import com.example.service.CourseService;
9
10 @RestController
11 @RequestMapping("/rest")
12 public class CourseRestController {
13     @Autowired
14     CourseService courseService;
15
16     @RequestMapping("/course/view/{id}")
17     public CourseModel view(@PathVariable(value = "id") String id) {
18         CourseModel course = courseService.selectCourse(id);
19         return course;
20     }
21 }
22 }
```

Penjelasan:

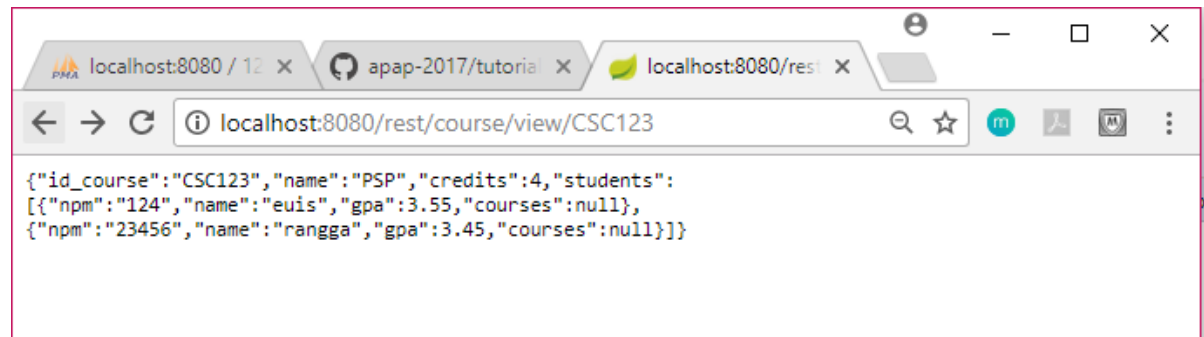
*Method view* dibuat agar pemanggilan *service Course view* dengan id tertentu dapat diakses dengan [localhost:8080/rest/course/view/CSC123](http://localhost:8080/rest/course/view/CSC123), dengan *mapping* `"/course/view/{id}"` akan dikembalikan dalam bentuk objek *CourseModel* ke halaman `viewCourse.html`

Atikah Luthfiana

1506689250

APAP B

Hasil :



Membuat *abstract method viewall* pada CourseService dan *implement method* tersebut pada CourseServiceDatabase:

```
1 package com.example.service;
2
3 import java.util.List;
4
5 import com.example.model.CourseModel;
6
7 public interface CourseService {
8     CourseModel selectCourse(String id);
9
10    List<CourseModel> selectAllCourses();
11 }
12
13
14
15
16
17
18
19
20
21
22
23 @Override
24 public List<CourseModel> selectAllCourses() {
25     Log.info("select all courses");
26     return courseMapper.selectAllCourses();
27 }
28
29
30
```

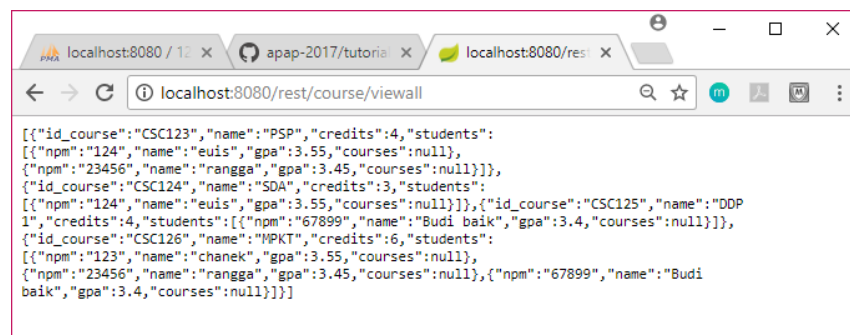
Membuat *method* selectAllCourses pada CourseMapper.java untuk mengambil data dari database:

```
24
25 @Select("select id_course, name, credits from course")
26 @Results(value = { @Result(property = "id_course", column = "id_course"), @Result(property = "name", column = "name"),
27     @Result(property = "credits", column = "credits"),
28     @Result(property = "students", column = "id_course", javaType = List.class, many = @Many(select = "selectStudent")) })
29 List<CourseModel> selectAllCourses();
30 }
31
```

Membuat *method* viewall pada CourseRestController untuk menampilkan semua query yang didapatkan:

```
24
25 @RequestMapping("/course/viewall")
26 public List<CourseModel> view(Model model) {
27     List<CourseModel> courses = courseService.selectAllCourses();
28     return courses;
29 }
30 }
31
```

Hasil:



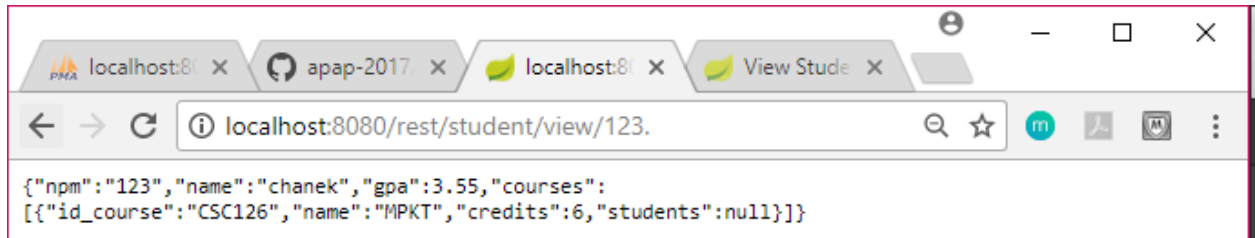
Atikah Luthfiana

1506689250

APAP B

## 2. Tutorial07Consumer

- Membuka halaman [localhost:8080/rest/student/view/123](http://localhost:8080/rest/student/view/123) dan [localhost:9090/student/view/123](http://localhost:9090/student/view/123)



- Implementasi *service customer* untuk view all Students (**Latihan 3**)  
Melengkapi *method* selectAllStudents pada StudentDAOImpl.java:

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

Melengkapi *method* selectAllStudents pada StudentServiceRest:

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

Penjelasan:

Method selectAllStudents di atas berfungsi untuk mengembalikan *query* "select \* from student" ke dalam objek *List* yang akan ditampilkan pada viewall.html

Atikah Luthfiana  
1506689250  
APAP B

Karena terjadi error pada bagian RestTemplate, maka ditambahkan *kode* berikut pada Tutorial07ConsumerApplication.java:

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

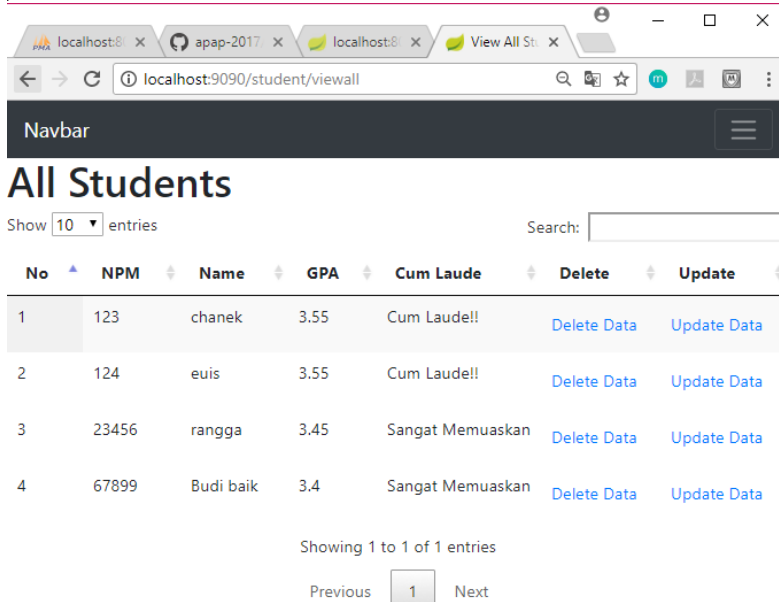
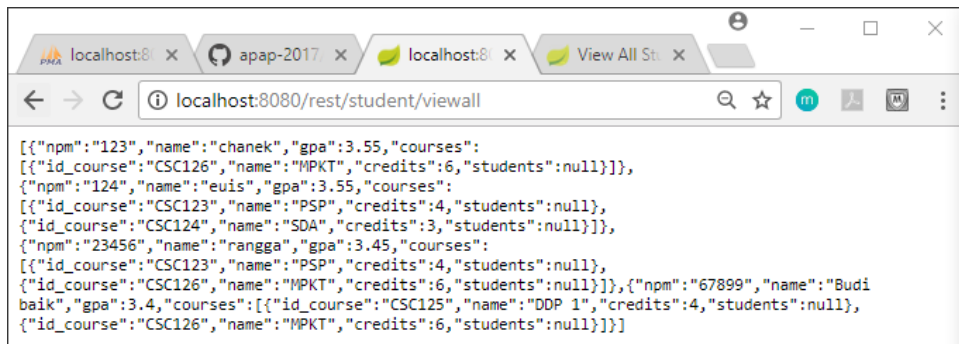
Compile Error yang muncul:

*No qualifying bean of type [org.springframework.web.client.RestTemplate] found for dependency [org.springframework.web.client.RestTemplate]: expected at least 1 bean which qualifies as autowire candidate for this dependency.*

Source: <https://stackoverflow.com/questions/36151421/could-not-autowire-fieldresttemplate-in-spring-boot-application>

Hasil :

2017-11-04 18:39:33.031 INFO 1896 --- [nio-9090-exec-1] com.example.service.StudentServiceRest : REST - select all students



Mata Kuliah APAP

Atikah Luthfiana

1506689250

APAP B

- Implementasi *service customer* untuk class CourseModel (**Latihan 4**)

Membuat class CourseDAO

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

Membuat class CourseDAOImpl

```
CourseDAO.java *CourseDAOImpl.java
1 package com.example.service;
2
3 import java.util.List;
4
5 @Service
6 public class CourseDAOImpl implements CourseDAO {
7     @Autowired
8     private RestTemplate restTemplate;
9
10    @Override
11    public CourseModel selectCourse(String id) {
12        CourseModel course = restTemplate.getForObject("http://localhost:8080/rest/course/view/" + id,
13            CourseModel.class);
14        return course;
15    }
16
17    @Override
18    public List<CourseModel> selectAllCourses() {
19        List<CourseModel> courses = restTemplate.getForObject("http://localhost:8080/rest/course/viewall", List.class);
20        return courses;
21    }
22 }
23
```

Penjelasan:

*Method* selectCourse (String id) berfungsi untuk mengembalikan pengambilan *course* berdasarkan *id* tertentu dan dikembalikan dalam objek CourseModel

Menambahkan abstract method selectAllCourses pada CourseService

```
CourseDAO.java CourseDAOImpl... CourseMappe... CourseServic...
1 package com.example.service;
2
3 import java.util.List;
4 import com.example.model.CourseModel;
5
6 public interface CourseService {
7     CourseModel selectCourse(String id);
8
9     List<CourseModel> selectAllCourses();
10 }
11
```

Atikah Luthfiana

1506689250

APAP B

### Membuat class CourseServiceRest

```
CourseDAO.java CourseDAOIm... CourseServ... CourseServ...
1 package com.example.service;
2
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 import lombok.extern.slf4j.Slf4j;
12
13 @Slf4j
14 @Service
15 @Primary
16 public class CourseServiceRest implements CourseService {
17     @Autowired
18     private CourseDAO courseDAO;
19
20     @Override
21     public CourseModel selectCourse(String id) {
22         log.info("REST - select course with id {}", id);
23         return courseDAO.selectCourse(id);
24     }
25
26     @Override
27     public List<CourseModel> selectAllCourses() {
28         log.info("REST - select all courses");
29         return courseDAO.selectAllCourses();
30     }
31 }
32
```

Penjelasan:

Pada kelas CourseServiceRest diimplementasikan *methods* yang ada pada CourseService yang mengembalikan fungsi berupa dijalankannya *methods* pada CourseDAO.

Menambahkan requestmapping untuk viewallcourses pada controller biasa dan rest controller pada service producer(telah dibuat sebelumnya)

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

    return "viewallcourses";
}

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

Atikah Luthfiana

1506689250

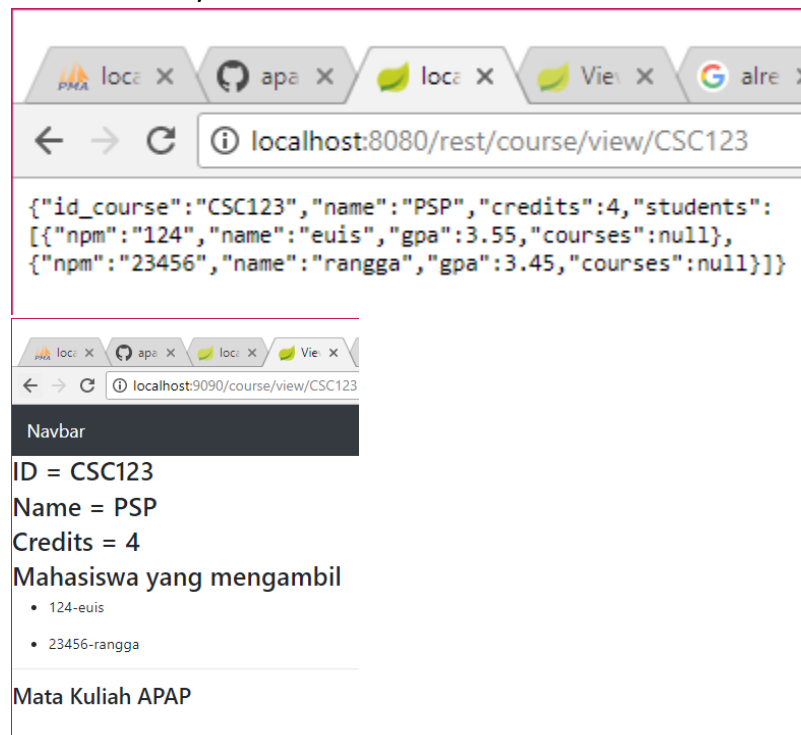
APAP B

## Menambahkan file viewallcourses.html

```
viewallcourses.html
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4 <title>View All Courses</title>
5 <link rel="stylesheet" href="/css/bootstrap.min.css" />
6 <link rel="stylesheet" type="text/css"
7 href="//cdn.datatables.net/1.10.16/css/jquery.dataTables.css"></link>
8 </head>
9
10 <body>
11 <div th:replace="fragments/fragment :: header"></div>
12 <h1>All Courses</h1>
13 <div>
14 <table id="table_id" class="display">
15 <thead>
16 <tr>
17 <th>No</th>
18 <th>ID</th>
19 <th>Name</th>
20 <th>Credits</th>
21 </tr>
22 </thead>
23 <tbody th:each="course, iterationStatus: ${courses}"
24 th:class="${iterationStatus.odd}? 'odd'">
25 <tr>
26 <td><p th:text="${iterationStatus.count}">No</p></td>
27 <td><p th:text="${course.id_course}">Course ID</p></td>
28 <td><p th:text="${course.name}">Course Name</p></td>
29 <td><p th:text="${course.credits}">Course Credits</p></td>
30 </tr>
31 </tbody>
32 </table>
33 </div>
34 <div th:replace="fragments/fragment :: footer"></div>
35 <script src="static/js/jquery-3.2.1.min.js"></script>
```

Hasil:

View course by Id



localhost:8080/rest/course/view/CSC123

```
{
  "id_course": "CSC123",
  "name": "PSP",
  "credits": 4,
  "students": [
    {
      "npm": "124",
      "name": "euis",
      "gpa": 3.55,
      "courses": null
    },
    {
      "npm": "23456",
      "name": "rangga",
      "gpa": 3.45,
      "courses": null
    }
  ]
}
```

localhost:9090/course/view/CSC123

Navbar

ID = CSC123  
Name = PSP  
Credits = 4  
Mahasiswa yang mengambil

- 124-euis
- 23456-rangga

Mata Kuliah APAP



Atikah Luthfiana

1506689250

APAP B

View all Courses

```
localhost:8080/rest/course/viewall

[{"id_course":"CSC123","name":"PSP","credits":4,"students":
[{"npm":"124","name":"euis","gpa":3.55,"courses":null},
{"npm":"23456","name":"rangga","gpa":3.45,"courses":null}],
{"id_course":"CSC124","name":"SDA","credits":3,"students":
[{"npm":"124","name":"euis","gpa":3.55,"courses":null}],{"id_course":"CSC125","name":"DDP
1","credits":4,"students":[{"npm":"67899","name":"Budi baik","gpa":3.4,"courses":null}],
{"id_course":"CSC126","name":"MPKT","credits":6,"students":
[{"npm":"123","name":"chanek","gpa":3.55,"courses":null},
{"npm":"23456","name":"rangga","gpa":3.45,"courses":null},{npm":"67899","name":"Budi
baik","gpa":3.4,"courses":null}]}
```

localhost:9090/course/viewall

Navbar

## All Courses

Show  entries

Search:

| No | ID     | Name  | Credits |
|----|--------|-------|---------|
| 1  | CSC123 | PSP   | 4       |
| 2  | CSC124 | SDA   | 3       |
| 3  | CSC125 | DDP 1 | 4       |
| 4  | CSC126 | MPKT  | 6       |

Showing 1 to 1 of 1 entries

Previous  Next

Mata Kuliah APAP

Atikah Luthfiana

1506689250

APAP B

Hal yang dipelajari pada tutorial 07:

- Membuat 2 buah *project* dalam satu folder yang selanjutnya keduanya dapat langsung di-*push* ke satu *repository* yang sama
- Tahap-tahap pembuatan *service producer*
- Tahap-tahap pembuatan *service consumer*
- Tahap-tahap menghubungkan *service producer* dan *service consumer* untuk keperluan *scalability*
- Mempelajari fungsi-fungsi: anotasi *RestController* untuk mengembalikan objek, mengembalikan objek dengan format JSON pada tampilan web, *RestTemplate* yang berfungsi untuk mengkonsumsi objek REST pada *web service*, *method* *getForObject*, dan anotasi *Primary* untuk menjalankan *Service* yang diinginkan; seperti *service* yang melakukan pengambilan data dari *web service*.