Atikah Zahrah Halim
1506721812
APAP – C

Latihan 1

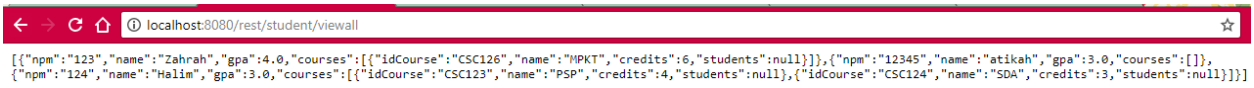
Buatlah service untuk mengembalikan seluruh student yang ada di basis data. Service ini mirip seperti method viewAll di Web Controller. Service tersebut di-mapping ke "/rest/student/viewall"

- Berikut merupakan method yang ditambahkan pada StudentRestController

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

Penjelasan: menambahkan request mapping ke /student/viewall kemudian membuat methodyang akan memanggil method selectAllStudents pada StudentService kemudian mengembalikannya.

- Hasil yang dikeluarkan



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).

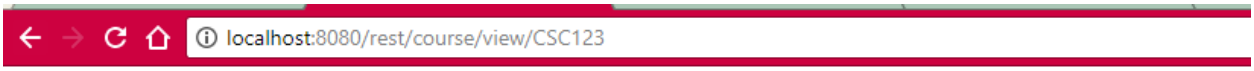
- Membuat CourseRestController pada package yang sama dengan StudentRestController kemudian masukkan implementasinya pada controller tersebut

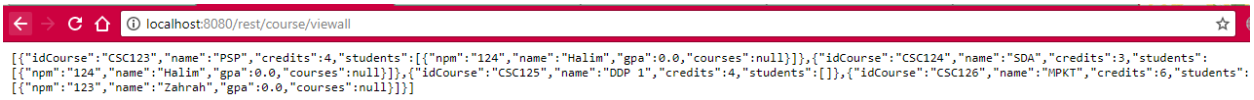
```
public class CourseRestController {
    @Autowired
    StudentService studentService;

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

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

Penjelasan: Untuk mendapatkan course dengan id tertentu, method yang dibuat akan ditambahkan request mapping ke course/view/{id} sesuai dengan idnya. Kemudian akan dikembalikan hasil dari pemanggilan method selectCourse dengan parameter id pada StudentService. Untuk mendapatkan semua course, method yang dibuat akan ditambahkan

request mapping course/viewall. Method tersebut akan mengembalikan hasil dari pemanggilan method selectAllCourse pada StudentService.

- Hasil yang dikeluarkan





Latihan 3

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

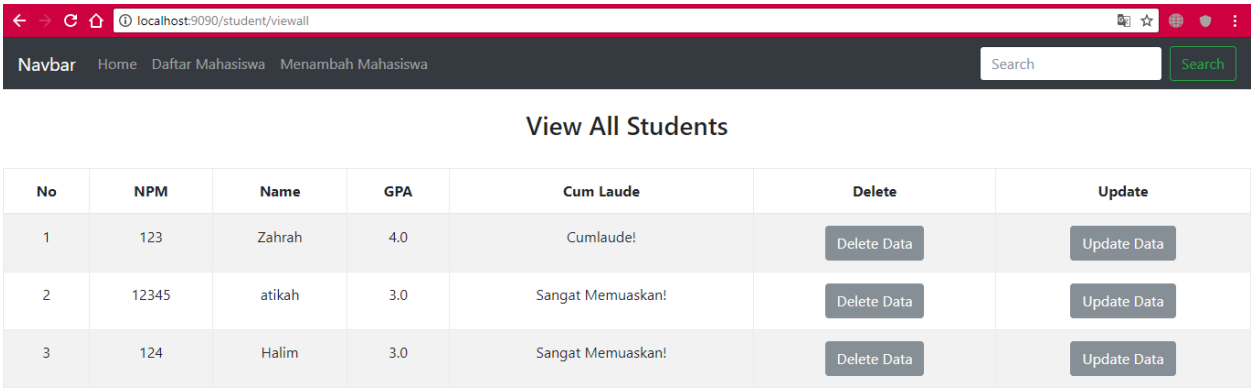
- Pada StudentDAO sudah terdapat method untuk selectAllStudents lalu implementasi pada StudentDAOImpl

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

- Pada StudentServiceRest diganti menjadi seperti berikut

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

- Hasil yang dikeluarkan



Atikah Zahrah Halim
1506721812
APAP – C
Latihan 4

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

- Membuat CourseService

```
public interface CourseService {
    CourseModel selectCourse (String id_course);

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

- Membuat CourseServiceRest

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

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

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

- Membuat CourseDAO

```
public interface CourseDAO {

    CourseModel selectCourse(String id);

    List<CourseModel> selectAllCourse();

}
```

Atikah Zahrah Halim
1506721812
APAP – C

- Membuat CourseDAOImpl

```
@Service
public class CourseDAOImpl implements CourseDAO {

    @Autowired
    private RestTemplate restTemplate;

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

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

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

- Mengimplementasikan viewall course pada controller

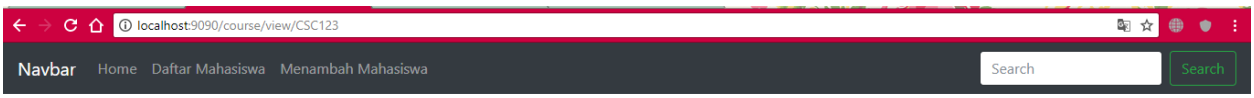
```
@RequestMapping("/course/view/{id_course}")
public String viewPath (Model model,
        @PathVariable(value = "id_course") String id_course)
{
    CourseModel course = courseDAO.selectCourse (id_course);

    if (course != null) {
        model.addAttribute ("course", course);
        return "view-course";
    } 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";
}
```

Atikah Zahrah Halim
1506721812
APAP – C

- Membuat viewall-course.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <title>View All Courses</title>
        <div th:replace="fragments/header"></div>
    </head>
    <body>
        <br></br>
        <center><h3>View All Courses</h3></center>
        <br></br>
        <table class="table table-bordered table-striped dt-responsive nowrap">
                <thead>
                    <tr>
                        <th class="text-center">No</th>
                        <th class="text-center">ID</th>
                        <th class="text-center">Name</th>
                        <th class="text-center">Credits</th>
                    </tr>
                </thead>
                <tbody>
                    <tr th:each="course,iterationStatus: ${courses}" th:class="${iterationStatus.odd}?'odd'">
                        <td class="text-center" th:text="${iterationStatus.count}">No.1</td>
                        <td class="text-center" th:text="${course.idCourse}">Course ID</td>
                        <td class="text-center" th:text="${course.name}">Course Name</td>
                        <td class="text-center" th:text="${course.credits}">Course Credits</td>
                    </tr>
                </tbody>
        </table>
            <script type="text/javascript" src="/js/datatables.min.js"></script>
        <script>
        $(document).ready( function () {
            $('#allStudents').DataTable();
        } );
        </script>

    </body>
    <br><br>
    <div th:replace="fragments/footer"></div>
</html>
```

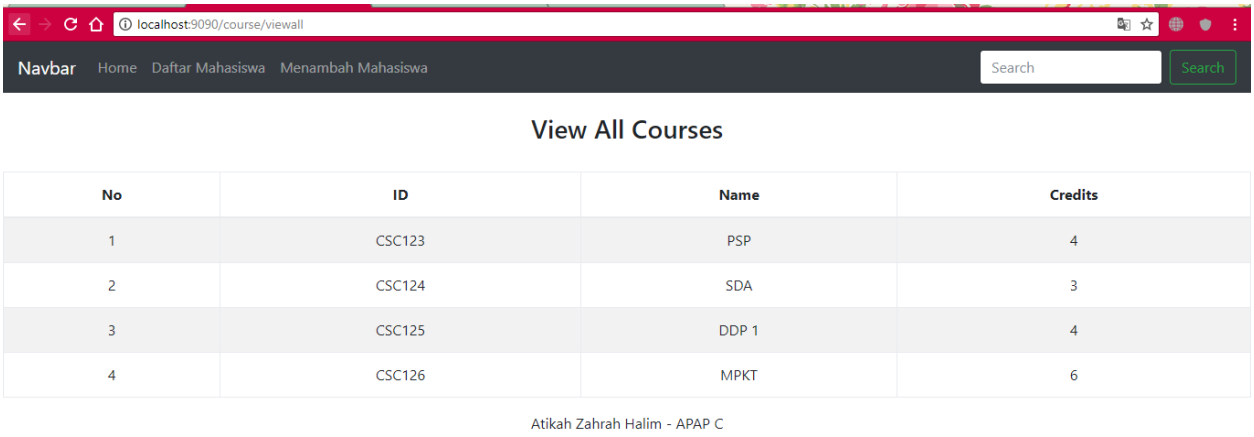
- Hasil yang dikeluarkan



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

- 124 - Halim

Atikah Zahrah Halim
1506721812
APAP – C



Lesson Learned

Di tutorial 7, saya belajar bagaimana cara menggunakan web service pada SpringBoot Framework. Untuk menjalankannya, diperlukan dua program yaitu Producer yang berfungsi untuk mengakses data langsung dari database dan mereturn objek masingmasing serta Consumer yang berfungsi untuk “menampilkan” fungsi tanpa perlu mengakses data secara langsung dari database dengan bantuan Producer dan REST. Pada Consumer, diperlukan DAO baru (interface dan implementasinya) untuk mengakses data melalui perantara Producer (tidak langsung dari database). Saya juga mempelajari anotasi baru yaitu @RestController dan @Primary