Anindito Izdihardian Wibisono
1506757466
APAP – C

Dalam tutorial kali ini, saya belajar mengenai penggunaan Thymeleaf sebagai template engine untuk proyek Spring Boot yang saya buat. Thymeleaf berfungsi untuk mencetak tampilan yang sesuai dengan hasil pemrosesan kode di Controller dan Model kedalam View yang ada. Tujuannya adalah memudahkan programmer menggunakan ulang struktur kode sebuah View secara berulang, dimana konten yang akan ditampilkan diisi oleh Thymeleaf. Selain itu, saya juga belajar mengenai penggunaan Bootstrap dan Datatables dalam mengembangkan view suatu proyek.

PERTANYAAN TUTORIAL

1. Nilai yang dihasilkan adalah “true” untuk indeks looping yang bernilai ganjil, dan “false” pada kasus lain.

2. th:unless memiliki fungsi sepadan dengan statement “else” pada bahasa Java, dan karenanya condition pada statement tersebut harus sama dengan condition pada statement th:if sebelumnya.

3. Terjadi 500 Internal Server Error dengan detail “Exception parsing document”. Hal ini dikarenakan terjadi perbedaan condition antara statement if-unless pada dokumen viewall.

4. Berikut potongan kode yang keluarannya sepadan dengan statement if-unless pada tutorial:

```
<td th:text="${student.gpa>=3.49}? 'Cum laude!' : 'Sangat Memuaskan!'">Cum Laude!</td>
```

5. Perintah th:replace mengganti seluruh isi elemen dengan fragment yang bersesuaian dari file html yang dispesifikasi. Misalnya, `th:replace=”fragments/fragment :: header”` akan mengganti isi tersebut dengan potongan kode berlabel th:fragment ‘header’ pada file fragment.html yang tersimpan pada folder fragments.

6. Ya, metode ini dapat dilakukan juga untuk kode 500. Metode ini bahkan bisa digunakan untuk mengganti kode error lain seperti 502 Bad Gateway, asalkan file html yang menangani error tersebut bernama tepat sama seperti kode error yang ingin digantikan halamannya.

LATIHAN 1: IMPLEMENTASI DATATABLE

```
<body>
    <h1>All Students</h1>
    <table id="studentData" class="display cell-border" width="100%"
        cellspacing="0">
        <thead>
            <tr>
                <th>No.</th>
                <th>NPM</th>
                <th>Nama</th>
                <th>GPA</th>
                <th>Status kelulusan</th>
                <th>Operasi</th>
            </tr>
        </thead>
        <tbody>
            <tr th:each="student,iterationStatus: ${students}">
                <td th:text="${iterationStatus.count}"></td>
                <td th:text="${student.npm}">Student NPM</td>
                <td th:text="${student.name}">Student Name</td>
                <td th:text="${student.gpa}">Student GPA</td>
                <td
                    th:text="${student.gpa>=3.49}? 'Cum laude!' : 'Sangat Memuaskan!'">Cum
                    Laude!</td>
                <td><a th:href="'/student/delete/' + ${student.npm}">Delete
                        Data</a> <br /> <a th:href="'/student/update/' + ${student.npm}">Update
                        Data</a></td>
            </tr>
        </tbody>
    </table>
    <script>
        $(document).ready(function() {
            $('#studentData').DataTable();
        });
    </script>
</body>
```

Screenshot potongan kode yang relevan untuk pengerjaan latihan ini. Perlu diperhatikan bahwa di head html, file css dan js dari datatables sudah diimpor dengan menggunakan CDN.

All Students

Show 10 entries

Search:

No.	NPM	Nama	GPA	Status kelulusan	Operasi
1	1001	Shen	3.98	Cum laude!	Delete Data Update Data
7	1007	Roy	3.91	Cum laude!	Delete Data Update Data
4	1004	Jasper	3.88	Cum laude!	Delete Data Update Data
8	1008	Jerry	3.87	Cum laude!	Delete Data Update Data
3	1003	Genji	3.75	Cum laude!	Delete Data Update Data
10	1010	Lena	3.49	Cum laude!	Delete Data Update Data
6	1006	Ninian	3.48	Sangat Memuaskan!	Delete Data Update Data
9	1009	Patrick	3.45	Sangat Memuaskan!	Delete Data Update Data
2	1002	Bradford	3.44	Sangat Memuaskan!	Delete Data Update Data
5	1005	Rachel	3.14	Sangat Memuaskan!	Delete Data Update Data

Showing 1 to 10 of 10 entries

Previous 1 Next

Screenshot berupa hasil keluaran pada browser. Pada screenshot ini, data diurutkan berdasarkan besaran GPA dari tertinggi hingga terendah dengan menggunakan tombol sorting pada kolom GPA.

LATIHAN 2: IMPLEMENTASI FRAGMENT DINAMIS

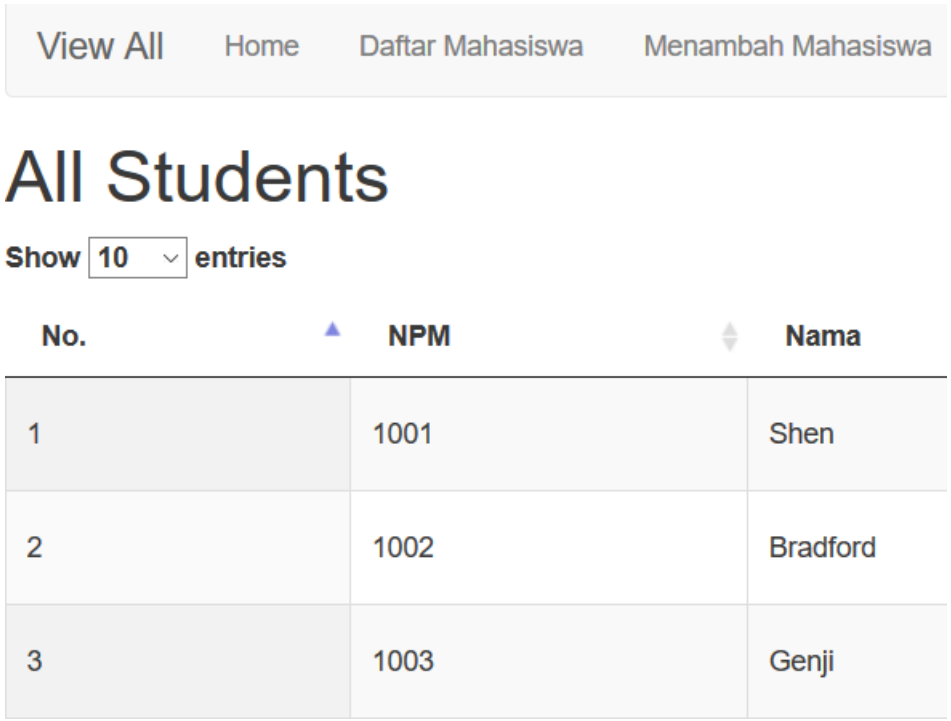
```
<div th:fragment="header">
  <nav class="navbar navbar-default">
  <div class="container-fluid">
  <div class="navbar-header">
    <a th:text="${title}" class="navbar-brand" href="#">NavBar</a>
  </div>
  <ul class="nav navbar-nav">
  <li class="nav-item"><a class="nav-link" href="/">Home</a></li>
  <li class="nav-item"><a class="nav-link" href="/student/viewall">Daftar Mahasiswa</a></li>
  <li class="nav-item"><a class="nav-link" href="/student/add">Menambah Mahasiswa </a></li>
  </ul>
</div>
```

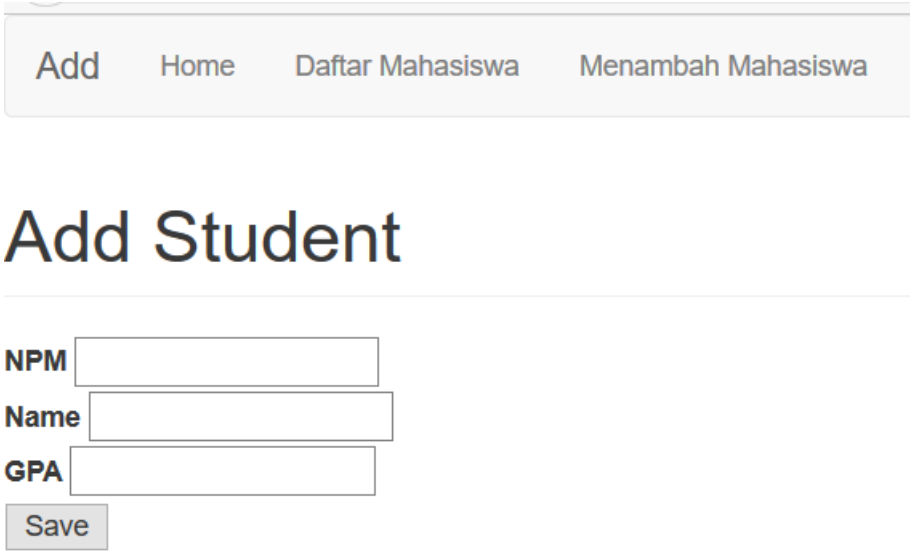
Screenshot potongan kode yang ditulis di fragment.html. Perhatikan atribut th:text yang ditambahkan.

```
@RequestMapping("/")
public String index (Model model)
{
    model.addAttribute ("title", "Homepage");
    return "index";
}


@RequestMapping("/student/add")
public String add (Model model)
{
    model.addAttribute ("title", "Add");
    return "form-add";
}
```

Screenshot sebagian perubahan kode pada StudentController.java. Disini, semua method ditambahkan parameter model agar method addAttribute untuk mensubstitusi teks pada fragment dapat dipanggil.





Screenshot berisi hasil keluaran pada browser. Disini, terlihat bahwa judul pada navbar berubah secara dinamis. Screenshot kiri diambil di halaman /viewall, sementara screenshot kanan diambil di halaman /student/add.