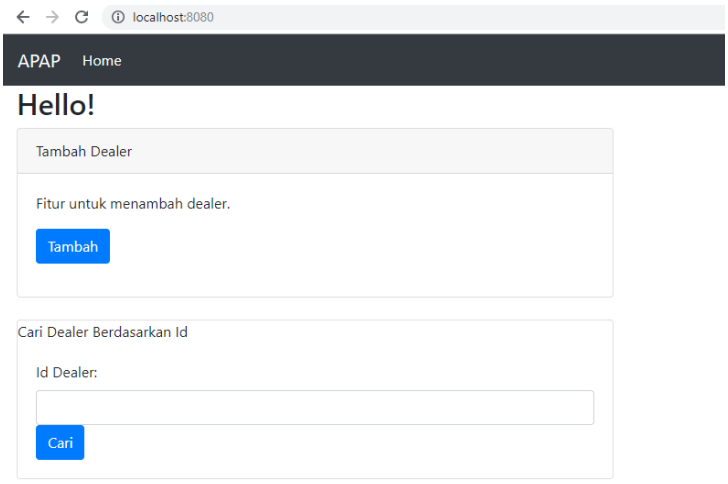
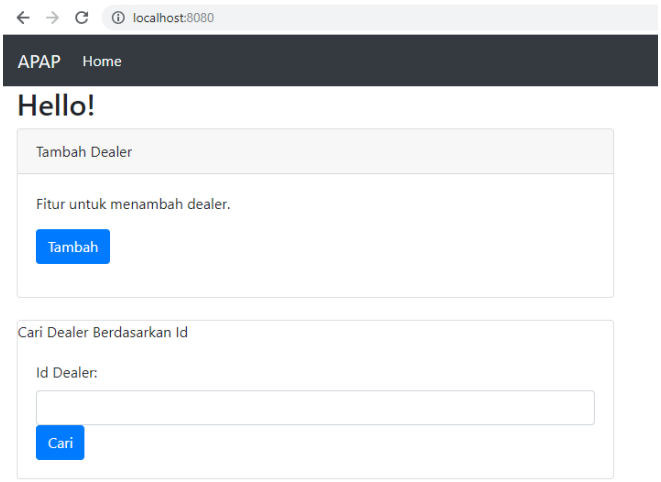
TUTORIAL

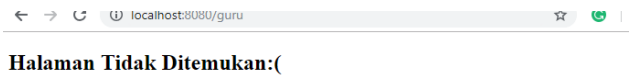
Home1



Home2 fragment



Error handler 404



LATIHAN

1. Pada tutorial, Anda sudah mencoba membuat fragment navigation bar. Tetapi, navigation bar pada tutorial tersebut tidak dinamis. Data HTML pada header akan sama di setiap view yang memakai fragment tersebut.

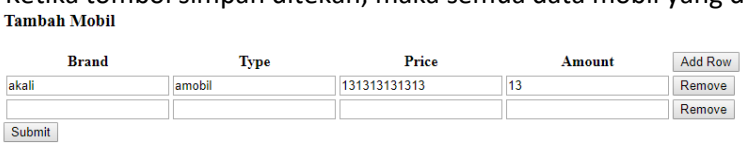
   Pada latihan kali ini, Anda diminta untuk membuat sebuah fragment header yang dinamis. Dimana, aspek dinamis tersebut terdapat pada title yang diletakkan pada fragment header, bukan setiap page HTML yang ada. Lebih jauh, title tersebut akan berubah-ubah sesuai dengan page yang ada. Silahkan implementasi dengan kebutuhan tersebut, dan jika sudah berhasil, pastikan header tersebut dimuat pada setiap view yang ada di project ini.

   Fragment.html modifikasi:
   ```
   <title th:text= "${title}"></title>
   ```

   Pada setiap target page, buat object baru dengan atribut fragmet inculde
   ```
   <object th:include="fragments/fragment :: title" th:remove="tag"></object>
   ```

   Lakukan pemanggilan target fragment pada controller, dan masukkan sesuai kebutuhan, contoh:
   ```
   model.addAttribute("title", "Add Car");
   ```

2. Pada bagian form handler, anda telah mempelajari cara meng-handle input multiple checkbox untuk dapat melakukan delete banyak mobil sekaligus.
   Pada latihan ini, anda diminta untuk dapat melakukan insert banyak mobil sekaligus. Salah satu contoh implementasinya sebagai berikut:
   -Pada halaman add-car terdapat tombol untuk menambah field untuk object mobil lainnya
   -Ketika tombol tambah ditekan, maka akan muncul field untuk insert mobil lainnya, dan ketika tombol hapus ditekan, maka row tersebut akan terhapus.
   -Ketika tombol simpan ditekan, maka semua data mobil yang diinput akan ter-insert ke database

Modifikasi addCar:

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <object th:include="fragments/fragment :: title" th:remove="tag"></object>
        <object th:include="fragments/fragment :: css" th:remove="tag"></object>
        <object th:include="fragments/fragment :: js" th:remove="tag"></object>
    </head>

    <body>
    <nav th:replace="fragments/fragment :: navbar"></nav>
        <h2>Hello!</h2>

        <h3>Tambah Mobil</h3>
        <form th:action="@{/car/add/} + ${dealer.id}" th:object="${dealer}" method = "POST">
        <input type="hidden" th:field="*{id}"></input>

            <table>
            <thead>
                <tr>
                    <th>Brand</th>
                    <th>Type</th>
                    <th>Price</th>
                    <th>Amount</th>
                    <th><button type="submit" name="addRow">Add Row</button></th>
                </tr>
            </thead>
            <tbody>
            <tr th:each="car,rowStat : *{listCar}">
                <td><input type="text" th:field="*{listCar[__${rowStat.index}__].brand}"/></td>
                <td><input type="text" th:field="*{listCar[__${rowStat.index}__].type}"/></td>
                <td><input type="number" th:field="*{listCar[__${rowStat.index}__].price}"/></td>
                <td><input type="number" th:field="*{listCar[__${rowStat.index}__].amount}"/></td>
                <td><button type="submit" name="removeRow" th:value="${rowStat.index}">Remove</button></td>
            </tr>
            </tbody>
        </table>
        <button type="submit" name="save">Submit</button>
```

Fungsi penambahanan row pada car controller:

```
@RequestMapping(value = "/car/add/{dealerId}", params= {"addRow"}, method =
RequestMethod.POST)
private String addRow (@ModelAttribute DealerModel dealer, Model model) {

       dealer.getListCar().add(new CarModel());
       model.addAttribute("dealer", dealer);
       return "addCar";
       }
```

Fungsi penghapusan row pada car controller:

```
@RequestMapping(value="/car/add/{dealerId}", method = RequestMethod.POST,
params={"removeRow"})
private String removeRow (@ModelAttribute DealerModel dealer, final
BindingResult bindingResult, final HttpServletRequest req, Model model) {
       final Integer rowId = Integer.valueOf(req.getParameter("removeRow"));
       dealer.getListCar().remove(rowId.intValue());
       model.addAttribute("dealer", dealer);
       return "addCar";
       }
```

Perubahan pada fungsi add car controller:

```
@RequestMapping(value = "/car/add/{dealerId}", method = RequestMethod.GET)
private String add(@PathVariable(value = "dealerId") Long dealerId, Model
model) {
       DealerModel dealer = dealerService.getDealerDetailById(dealerId).get();
```

```
ArrayList<CarModel> list = new ArrayList<CarModel>();
list.add(new CarModel());
dealer.setListCar(list);
model.addAttribute("title", "Add Car");
model.addAttribute("dealer", dealer);
return "addCar";
}
```

Fungsi addCarSubmit untuk penyimpan car sesuai id dealer:

```
@RequestMapping(value = "/car/add/{dealerId}", method = RequestMethod.POST,
params= {"save"})
private String addCarSubmit (@ModelAttribute DealerModel dealer) {
	DealerModel dealerAcv =
	dealerService.getDealerDetailById(dealer.getId()).get();
		for(CarModel car : dealer.getListCar()) {
			car.setDealer(dealerAcv);
			carService.addCar(car);
			}
	return "add";
	}
```

3. Jelaskan perbedaan antara:
   - th:include > menyertakan isi fragmen ke dalam tag host-nya
     Atribut yang memasukkan bagian dari fragment ke dalam body dari host tag namun tidak memasukkan fragment tag.

   - th:replace > menggantikan tag host dengan fragmen.
     Atribut yang mensubstitusi host tag dengan menghapus isi host tag tersebut dan mengisi dengan bagian fragment termasuk juga fragment lainnya.