Ahmad Farhan H A
1506732766
APAP - A

Jawaban Pertanyaan

1.
2. Karena penggunaa GET akan menyebabkan data di-encoded dalam Request-URI. Banyak server dan pengguna akan melakukan log ke request URI sehingga dapat dilihat oleh pihak ketiga (https://stackoverflow.com/questions/3477333/what-is-the-difference-between-post-and-get). Selain itu penggunaan GET akan tersimpan di history bowser, bisa di bookmark, dan bisa di cache sehingga membuat penggunaan GET tidak aman (https://www.w3schools.com/TAgs/ref_httpmethods.asp). Perlu penanganan yang berbeda pada header method.

3. Tidak, akan menyebabkan exception. (https://stackoverflow.com/questions/17338828/can-i-use-both-post-and-get-on-the-same-method)

Method yang Saya buat pada Latihan Menambahkan Delete

```
@RequestMapping("/student/delete/{npm}")
public String delete (Model model, @PathVariable(value = "npm") String npm)
{
    StudentModel student = studentDAO.selectStudent (npm);

    if (student != null) {
        studentDAO.deleteStudent (npm);
        return "delete";
    } else {
        model.addAttribute ("npm", npm);
        return "not-found";
    }
}
```

Method delete yang saya buat memeriksa di database apakah terdapat student dengan npm sesuai parameter. Apabila terdapat student dengan npm sesuai parameter, maka student tersebut akan dihapus dari database dan mengembalikan String delete yang akan melempar ke page delete. Sedangkan apabila student dengan npm sesuai parameter tidak ditemukan, maka method ini akan membuat attribut npm sesuai parameter dan mengembalikan String not-found yang akan melemparkan ke page not-found.

Method yang Saya buat pada Latihan Menambahkan Update

```
@RequestMapping(value = "/student/update/submit", method = RequestMethod.POST)
public String updateSubmit (
        @RequestParam(value = "npm", required = false) String npm,
        @RequestParam(value = "name", required = false) String name,
        @RequestParam(value = "gpa", required = false) double gpa)
{

    studentDAO.updateStudent (npm, name, gpa);

     return "success-update";
}
```

Method yang saya buat menerima input parameter npm, name , dan gpa. Kemudian method akan melakukan update pada database dan mengubah data pada student dengan npm yang sama dengan parameter npm.

Method yang Saya buat pada Latihan Menggunakan Object Sebagai Parameter

```
@RequestMapping(value = "/student/update/submit", method = RequestMethod.POST)
public String updateSubmit (StudentModel student)
{

    studentDAO.updateStudent (student);

     return "success-update";
}
```

Method ini hampir sama dengan method yang telah dibuat sebelumnya. Perbedaanya terletak pada perubahan parameter, dimana parameter yang digunakan sekarang berupa StudentModel. Dengan perubahan parameter pada method updateSubmit kita juga harus merubah parameter di StudentService, StudentServiceDatabase, dan StudentMapper.