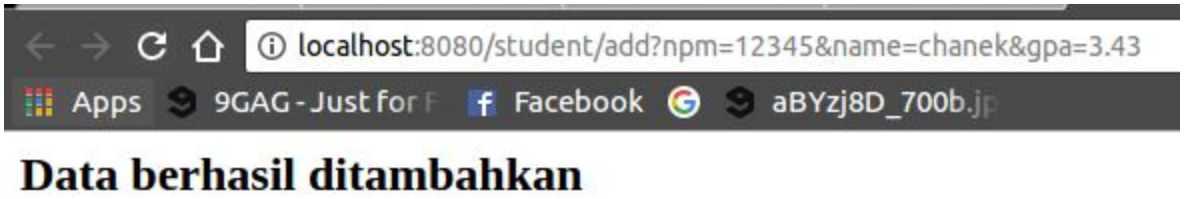
localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43
Pertanyaan 1: apakah hasilnya? Jika error, tuliskan penjelasan Anda.

Tidak error, data berhasil ditambahkan ke List.



Data berhasil ditambahkan

localhost:8080/student/add?npm=12345&name=chanek
Pertanyaan 2: apakah hasilnya? Jika error, tuliskan penjelasan Anda.

Error, karena tidak terdapat parameter ‘gpa’ sementara di program ‘gpa’ bersifat required.

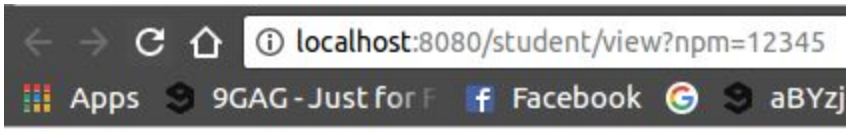
Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Wed Sep 20 21:01:43 WIB 2017
There was an unexpected error (type=Bad Request, status=400).
Required double parameter 'gpa' is not present

localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43 lalu buka
localhost:8080/student/view?npm=12345
Pertanyaan 3: apakah data Student tersebut muncul? Jika tidak, mengapa?

Muncul informasi student dengan npm yang sesuai.



NPM = 12345

Name = chanek

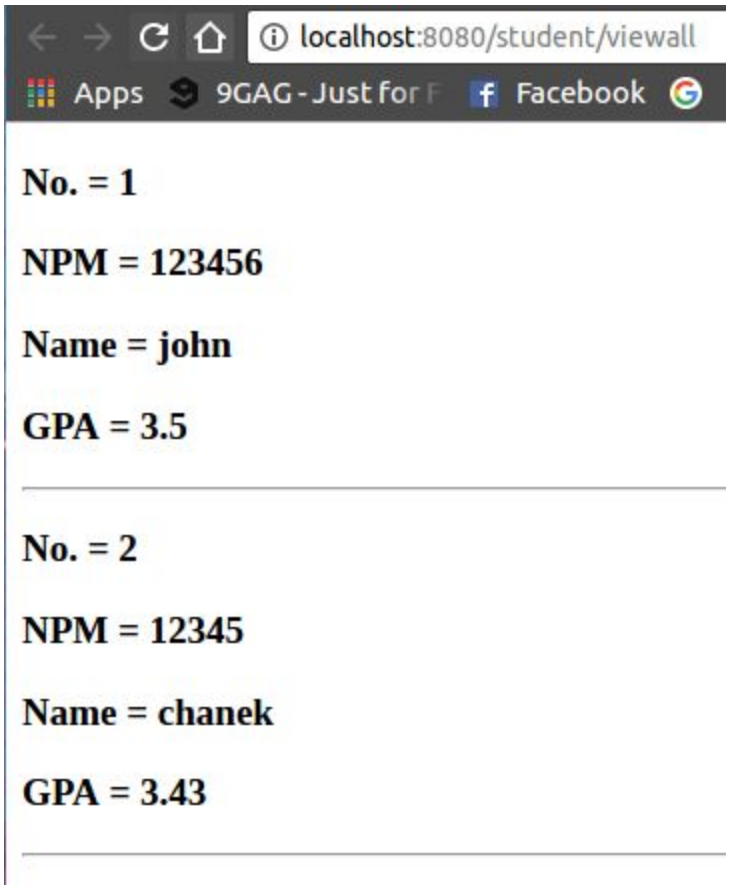
GPA = 3.43

Coba matikan program dan jalankan kembali serta buka
localhost:8080/student/view?npm=12345
Pertanyaan 4: apakah data Student tersebut muncul? Jika tidak, mengapa?

Tidak muncul karena program sudah di stop terlebih dahulu dan data Student juga ikut terhapus.

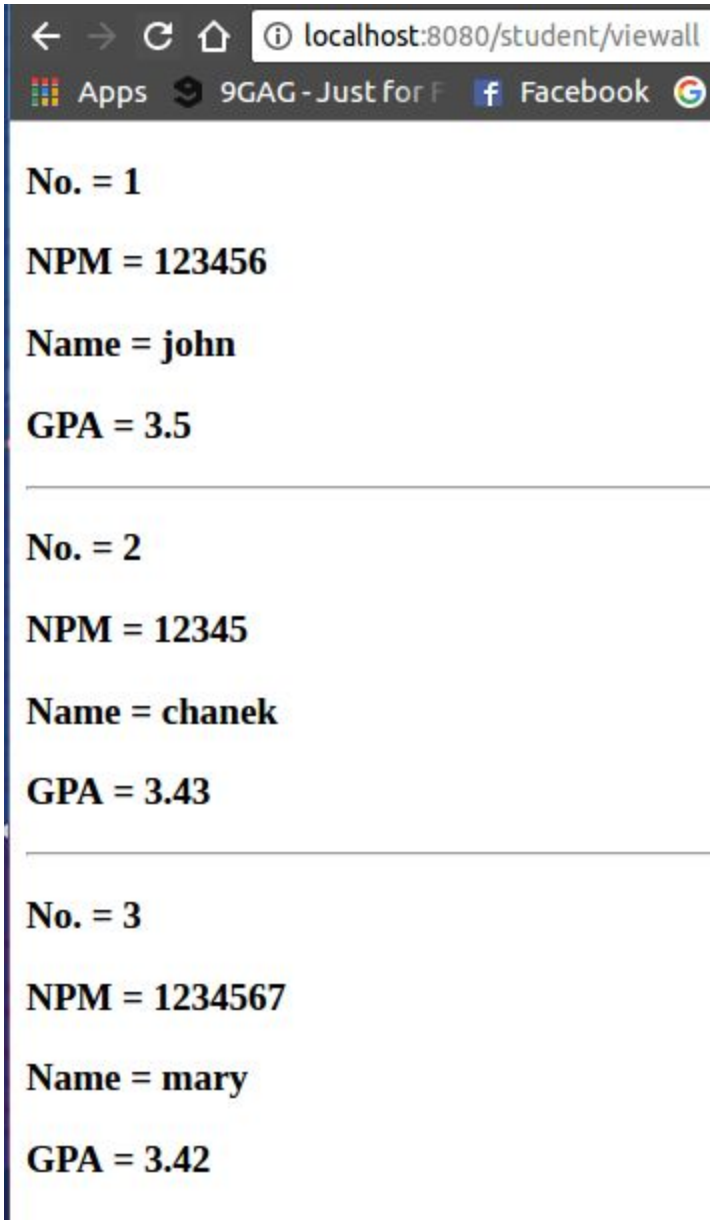
localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43 lalu buka
localhost:8080/student/viewall,
Pertanyaan 5: apakah data Student tersebut muncul?

Data tersebut muncul beserta dengan data Student yang sudah ditambahkan sebelumnya.



Coba tambahkan data Student lainnya dengan NPM yang berbeda, lalu buka localhost:8080/student/viewall,
Pertanyaan 6: Apakah semua data Student muncul?

Data tersebut muncul.



Method selectStudent

```
@Override
public StudentModel selectStudent(String npm) {
    for(int i = 0; i < studentList.size(); i++) {
        if(npm.equals(studentList.get(i).getNpm())) {
            return studentList.get(i);
        }

    }
    return null;
}
```

Pertama-tama kita menelusuri seluruh isi dari list-list student, kemudian dibandingkan apakah ada npm yang sama dengan input, jika ada maka object Student tersebut dikembalikan, jika tidak ada maka null dikembalikan.

Penjelasan method delete

Pertama-tama membuat method deleteStudent() terlebih dahulu di StudentService dengan menerima input parameter npm. Setelah itu, kita implementasikan method tersebut di InMemoryStudentService

```
@Override
public void deleteStudent(String npm) {
    StudentModel selectedStudent = selectStudent(npm);
    studentList.remove(selectedStudent);
}
```

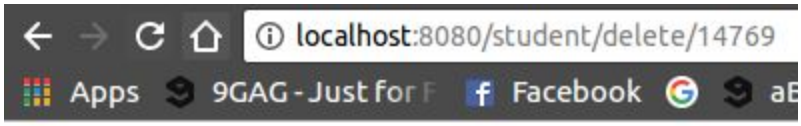
Prosesnya adalah kita mencari terlebih dahulu Student tessebut berdasarkan npmnya dengan menggunakan method selectStudent() yang sudah diimplementasikan sebelumnya. Setelah itu, kita menghapus selectedStudent tersebut. Kemudian kita harus membuat controller untuk method delete tersebut di class StudentController

```
@RequestMapping(value = {"/student/delete/{npm}"})
public String deletePath(@PathVariable String npm, Model model) {
    model.addAttribute("npm", npm);
    StudentModel student = studentService.selectStudent(npm);
    if(student != null) {
        studentService.deleteStudent(npm);
        return "delete";
    }
    return "error";
}
```

Setelah itu, kita membutuhkan file template html, sehingga kita harus membuat file delete.html yang nantinya akan menjadi tampilan di browser.

```
1 <html>
2 <head>
3  <title>Delete</title>
4  </head>
5 <body>
6      <h2>Data berhasil dihapus</h2>
7  </body>
8  </html>
```

Setelah itu kita jalankan di browser, maka akan keluar tampilan seperti berikut.



Data berhasil dihapus