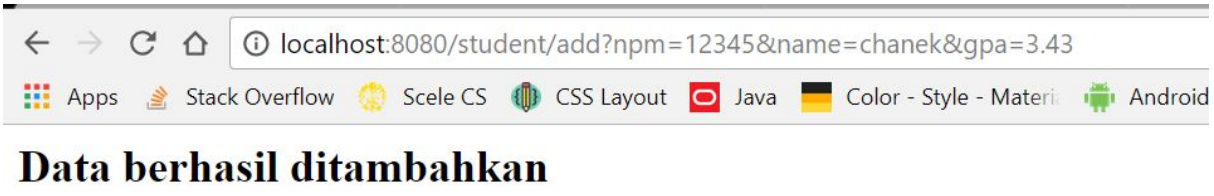
Nama: Haryo Parigroho
NPM: 1506689603
Kelas: B

A. Ringkasan materi

Yang saya pelajari dari tutorial kali ini adalah konsep mvc yang lebih mendalam dari tutorial sebelumnya. Pada tutorial kali ini sudah menggunakan model dan juga service. Model adalah class yang merepresentasikan suatu objek di database yang menyimpan informasi tertentu (atribut dan method), service adalah sebuah interface yang menghubungkan controller dengan database (pada tutorial kali ini menggunakan ArrayList of student). Service juga merupakan class yang melakukan pengolahan data dari database.

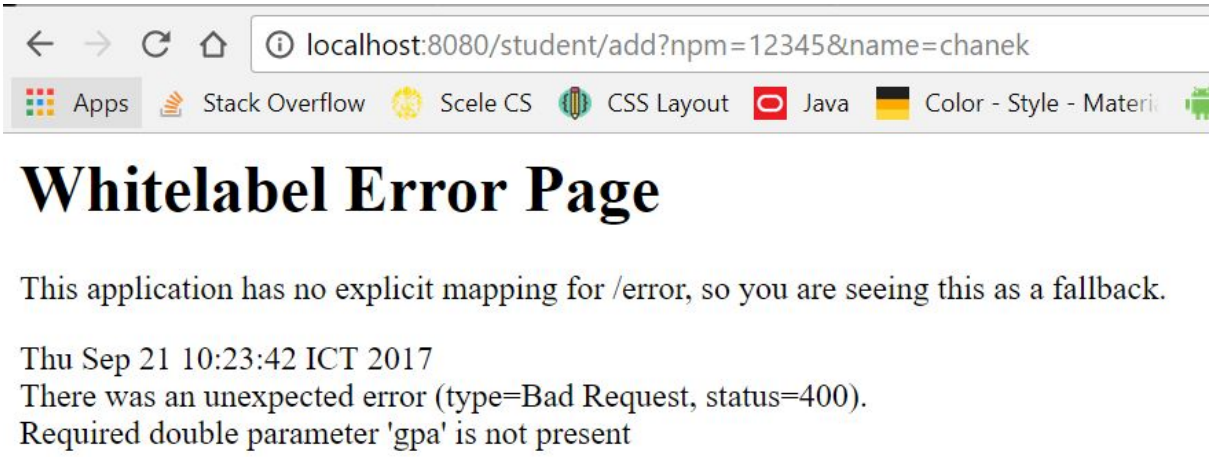
B. Jawaban Tutorial

Pertanyaan 1



Hasilnya menampilkan “Data berhasil ditambahkan” berarti *student* dengan nama chanek, npm = 12345 yang memiliki gpa 3.43 berhasil ditambahkan ke dalam *ArrayList*.

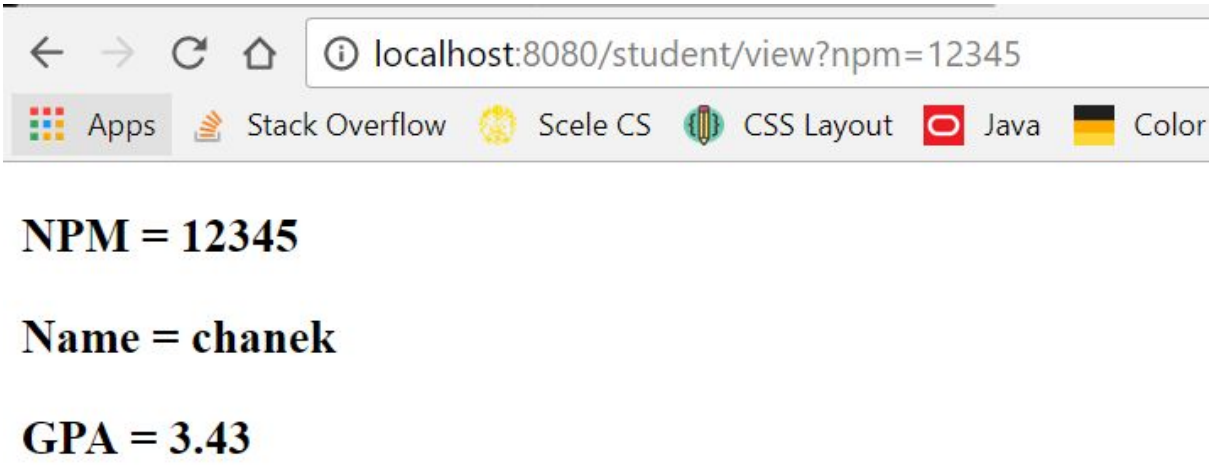
Pertanyaan 2



Terjadi error karena *RequestMapping* /student/add juga meminta parameter gpa dengan *attribute required=true* berarti harus ada parameter gpa pada GET request.

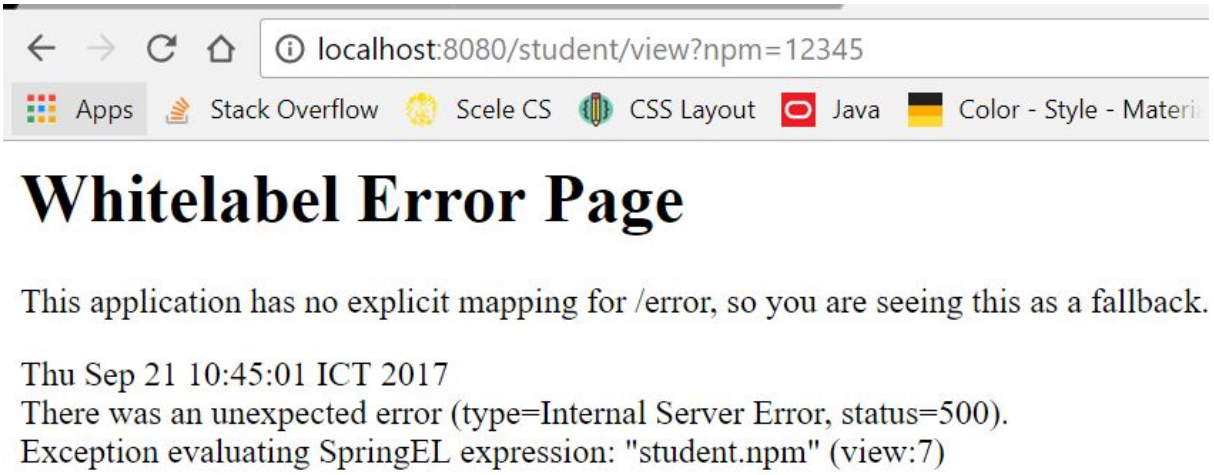
Nama: Haryo Parigroho
NPM: 1506689603
Kelas: B

Pertanyaan 3



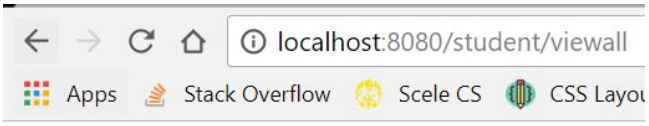
Data *student* muncul.

Pertanyaan 4



Data *student* tidak muncul karena *student* dengan npm 12345 tidak ada di dalam ArrayList setelah program dimatikan.

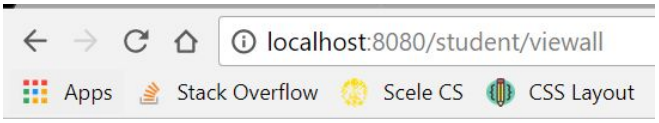
Pertanyaan 5



Data *student* muncul.

Nama: Haryo Parigroho
NPM: 1506689603
Kelas: B

Pertanyaan 6



No. 1

NPM = 12345

Name = chanek

GPA = 3.43

No. 2

NPM = 67890

Name = haryo

GPA = 4.0

Semua data student muncul karena menggunakan th:each Thymeleaf untuk melakukan iterasi pada list students.

C. Method selectStudent

```
@Override
public StudentModel selectStudent(String npm) {
	// TODO Auto-generated method stub
	for(StudentModel student : studentList) {
		if(student.getNpm().equals(npm)) {
			return student;
		}
	}
	return null;
}
```

D. Penjelasan Tahap-Tahap Mengerjakan Latihan

1. Membuat method view student dengan menggunakan PathVariable.
   Tahap-tahap yang saya lakukan untuk menyelesaikan method ini antara lain; Mengecek apakah ada *student* yang memiliki npm itu jika ada maka menambahkan atribut student dengan status ‘ok’ ke dalam model tetapi jika tidak ada atau npm tidak diberikan maka hanya menambahkan atribut status ‘notfound’. Setelah methodnya selesai kemudian saya memodifikasi template view.html menjadi menggunakan ‘switch’ milik thymeleaf, jika status ‘ok’ maka

menampilkan data *student* tetapi jika status ‘notfound’ maka menampilkan informasi data tidak ditemukan.

2. Membuat method delete student dengan menggunakan PathVariable. Tahap-tahap yang saya lakukan untuk menyelesaikan method ini antara lain; Membuat method deleteStudent yang mengembalikan student yang dihapus pada interface StudentService lalu mengimplementasikan method tersebut di InMemoryStudentService. Setelah itu pada method deleteStudent pada controller, method deleteStudent pada service dipanggil, jika method mengembalikan student makan menambahkan atribut student pada model dan status ‘ok’ tetapi jika method mengembalikan null maka hanya menambahkan atribut status ‘notfound’ pada model. Setelah selesai kemudian saya membuat template delete.html yang menggunakan ‘switch’ miliki thymeleaf, jika status ‘ok’ maka menampilkan informasi bahwa data berhasil dihapus dan menampilkan informasi *student* yang dihapus tetapi jika status ‘notfound’ maka menampillkan informasi data tidak ditemukan.