

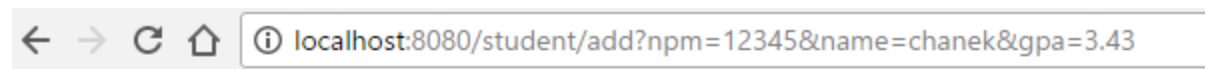
Ringkasan dari materi yang Anda telah pelajari pada tutorial kali ini

Pada tutorial kali ini mempelajari bagaimana penggunaan model pada Spring boot yang mencakup relasi dan implementasinya terhadap kontroller dan view.

Hasil jawaban dari setiap poin pada bagian tutorial (dapat didukung dengan screenshot)

localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43

Pertanyaan 1: apakah hasilnya? Jika error, tuliskan penjelasan Anda.

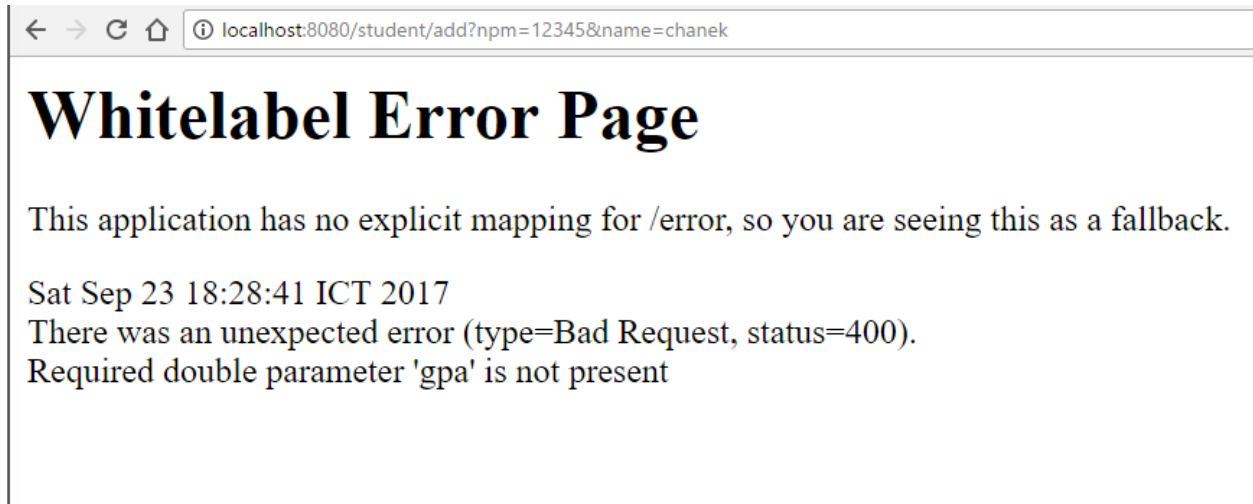


Data berhasil disimpan

Tidak error karena semua variabel terdapat pada parameter di url, sehingga variabel tersebut bisa diproses pada kontroller.

localhost:8080/student/add?npm=12345&name=chanek

Pertanyaan 2: apakah hasilnya? Jika error, tuliskan penjelasan Anda.



Error karena tidak ada variable gpa dan setting pada kontroller mengharuskan adanya tiga variable yang salah satunya adalah GPA

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?



NPM = 12345

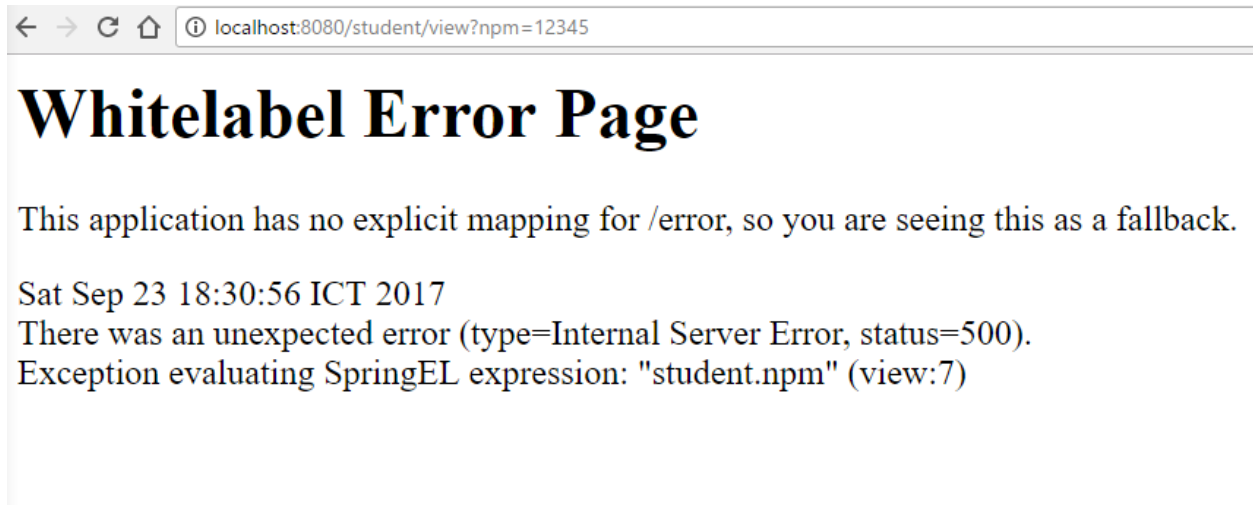
Name = chanek

GPA = 3.43

Data tersebut muncul

localhost:8080/student/view?npm=12345

Pertanyaan 4: apakah data Student tersebut muncul? Jika tidak, mengapa?



Data tidak muncul karena tidak adanya object dengan npm yang diminta.

localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43 lalu buka

localhost:8080/student/viewall,

Pertanyaan 5: apakah data Student tersebut muncul?



No. 1

NPM = 12345

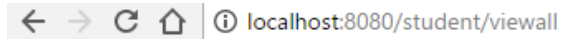
Name = chanek

GPA = 3.43

Muncul

localhost:8080/student/viewall,

Pertanyaan 6: Apakah semua data Student muncul?

A screenshot of a web browser's address bar. It features navigation icons (back, forward, refresh, home) on the left and an information icon on the right. The address bar contains the text 'localhost:8080/student/viewall'.

Karena isi dari StudentList kosong maka hanya mengembalikan halaman kosong.

Method selectStudent yang Anda implementasikan

```
@Override
public StudentModel selectStudent(String npm) {
    int totalStudents = studentList.size();

    for (int i = 0; i < totalStudents; i++) {
        if(studentList.get(i).getNpm().equals(npm)) {
            return studentList.get(i);
        }
    }
    return null;
}
```

Penjelasan fitur delete yang Anda buat pada bagian latihan.

Tahap yang saya lakukan untuk menambahkan fitur pada latihan:

Membuat method kosong pada interface StudentService

```
package com.example.tutorial3.service;

import com.example.tutorial3.model.StudentModel;

public interface StudentService {
    StudentModel selectStudent(String npm);
    List<StudentModel> selectAllStudents();
    void addStudent(StudentModel student);
    StudentModel deleteStudent(String npm);
}
```

Implementasi pada class InMemoryStduentService

```
@Override
public StudentModel deleteStudent(String npm) {
    int totalStudents = studentList.size();

    for (int i = 0; i < totalStudents; i++) {
        if (studentList.get(i).getNpm().equals(npm)) {
            StudentModel student = studentList.get(i);
            studentList.remove(i);
            return student;
        }
    }
    return null;
}
```


Membuat Controller delete dengan menggunakan Path variable

```
@RequestMapping(value = {"/student/delete","/student/delete/{npm}}")
public String deletePath(@PathVariable Optional<String> npm, Model model)
    StudentModel student = studentService.deleteStudent(npm.get());
    if(!npm.isPresent() || student.equals(null)) {
        return "errorDelete";
    }

    model.addAttribute("student",student);
    return "delete";
}
```

Menambahkan error page pada templates

```
1 <html>
2   <head>
3     <title>Error Delete</title>
4   </head>
5   <body>
6     <h2>nomor NPM kosong atau tidak ditemukan. Proses delete dibatalkan.
7   </h2>
8   </body>
9 </html>
```