Menggunakan Model dan Service pada Project Spring Boot

Tutorial ini akan mengimplementasikan konsep *model-view-controller* (MVC) menggunakan *Spring Boot* dengan fokus *model* dan *service*. Model pada tutorial in adalah objek mahasiswa yang memiliki nama, alamat, tanggal lahir, nomor pokok mahasiswa, dsbnya. Sementara untuk service adalah pengolahan meliputi kalkulasi data yang diambil dari database, manipulasi user input kedalam database, dsbnya.

Membuat Class Model

Pada **com.exampe.tutorial3.model** > StudentModel.java

Class ini untuk menyimpan variable name, npm dan gpa pada class Student

```
package com.example.tutorial3.model;

public class StudentModel {
	private String name;
	private String npm;
	private double gpa;

	public StudentModel (String name, String npm, double gpa) {
		this.name = name;
		this.npm = npm;
		this.gpa = gpa;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getNpm() {
		return npm;
	}

	public void setNpm(String npm) {
		this.npm = npm;
	}

	public double getGpa() {
		return gpa;
	}

	public void setGpa(double gpa) {
		this.gpa = gpa;
	}
}
```

Membuat Service

Pada **com.example.tutorial3.service** > StudentService.java

```
import java.util.List;

import com.example.tutorial3.model.StudentModel;

public interface StudentService {

	StudentModel selectStudent(String npm);

	List<StudentModel> selectAllStudents();

	void addStudent(StudentModel student);

}
```

Pada **com.example.tutorial3.service** > InMemoryStudentService.java

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

import java.util.ArrayList;

import java.util.List;

import com.example.tutorial3.model.StudentModel;

public class InMemoryStudentService implements StudentService{

	private static List<StudentModel> studentList = new
ArrayList<StudentModel>();

	@Override

	public StudentModel selectStudent(String npm) {

		for(StudentModel student: studentList) {

			if(student.getNpm().equals(npm)) {
				return student; }

		}

		return null;

	}

	@Override

	public List<StudentModel> selectAllStudents(){

		return studentList;

	}

	public void addStudent(StudentModel student) {

		studentList.add(student);}

	}
```

Nama : Lidia Natalia Manik
NPM : 1706106803

Membuat Controller dan Fungsi Add

Pada **com.example.tutorial3.controller** > StudentController.java

```
package com.example.tutorial3.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import com.example.tutorial3.service.InMemoryStudentService;
import com.example.tutorial3.service.StudentService;
import com.example.tutorial3.model.StudentModel;


@Controller
public class StudentController {
	private final StudentService studentService;

	public StudentController() {
		studentService = new InMemoryStudentService();
	}

	@RequestMapping("/student/add")
	public String add(@RequestParam(value = "npm", required = true) String npm,
					@RequestParam(value = "name", required =
true) String name,
					@RequestParam(value = "gpa", required =
true) double gpa) {

		StudentModel student = new StudentModel(npm, name, gpa);
		studentService.addStudent(student);

		return "add";
	}
}
```

Pada resources/templates > add.html

```
<html>
<head>
<title>Add</title>
</head>
<body>
	<h2> Data berhasil ditambahkan</h2>
</body>
</html>
```

Nama : Lidia Natalia Manik
NPM : 1706106803

Pertanyaan 1 : Apabila program dijalankan dan url berikut dibuka :

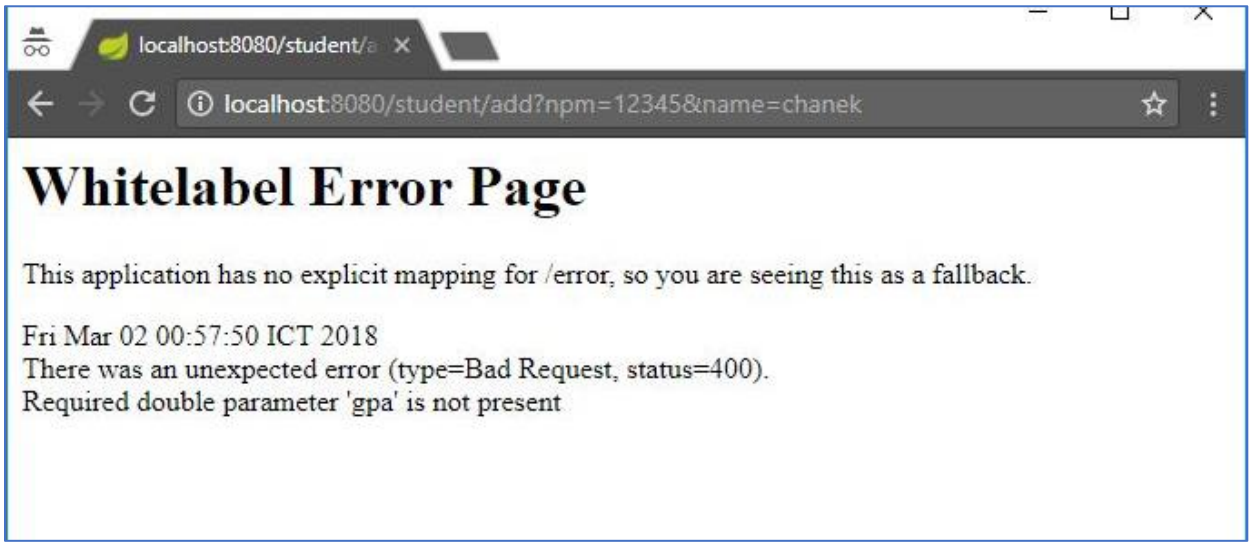
```
localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43
```

Hasilnya tidak error karena memiliki value pada masing-masing parameter npm, name dan gpa.

Pertanyaan 2 : Apabila program dijalankan dan url yang dibuka :

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

Hasilnya error karena parameter gpa tidak diisi value. Seharusnya gpa diisi nilai karena required.



Method View by NPM

Menambahkan method pada class StudentController

```
@RequestMapping("/student/view")
	public String view(Model model, @RequestParam(value = "npm", required =
true) String npm) {
		StudentModel student = studentService.selectStudent(npm);
		model.addAttribute("student", student);
		return "view";
	}
```

Pada resources/templates > view.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>View Student by NPM</title>
</head>
<body>
		<h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
		<h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
		<h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
</body>
</html>
```

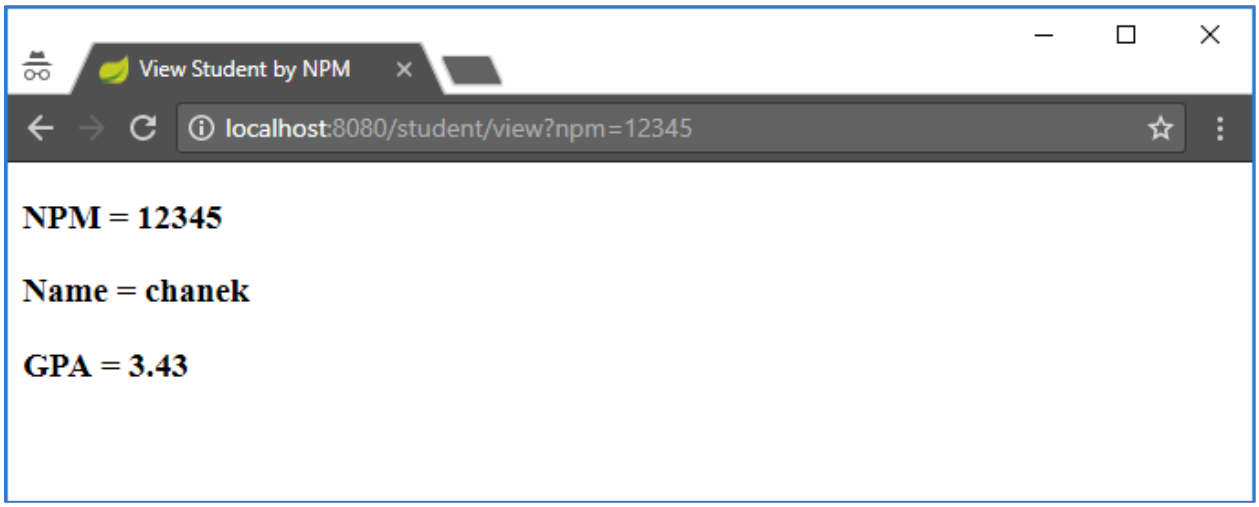
Pertanyaan 3 : Apabila program dijalankan dan url yang dibuka :

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

Kemudian dijalankan

```
localhost:8080/student/view?npm=12345
```

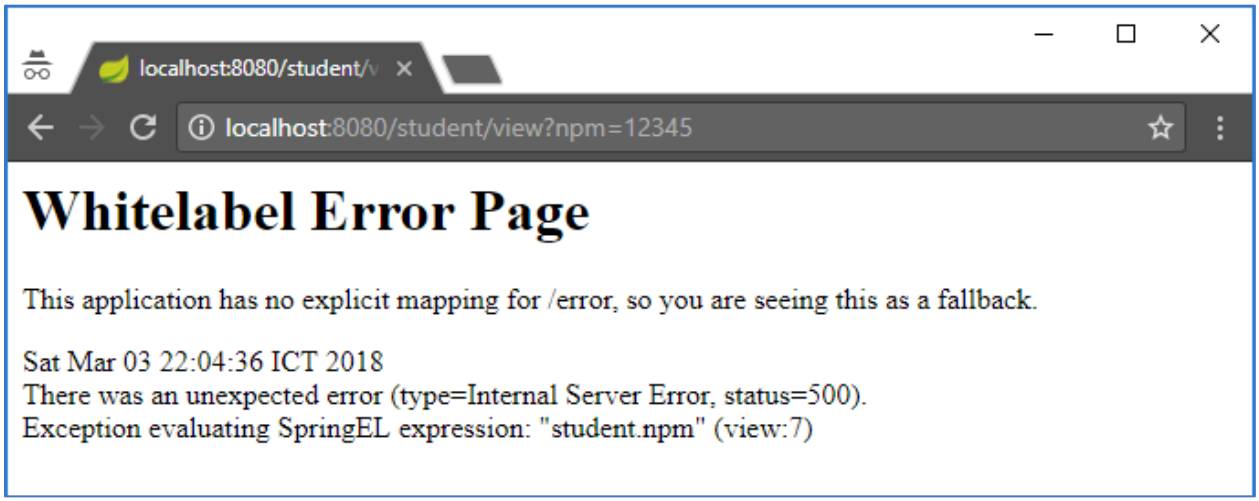
Hasilnya : page akan menampilkan data student berdasarkan parameter yang dimasukkan di url.



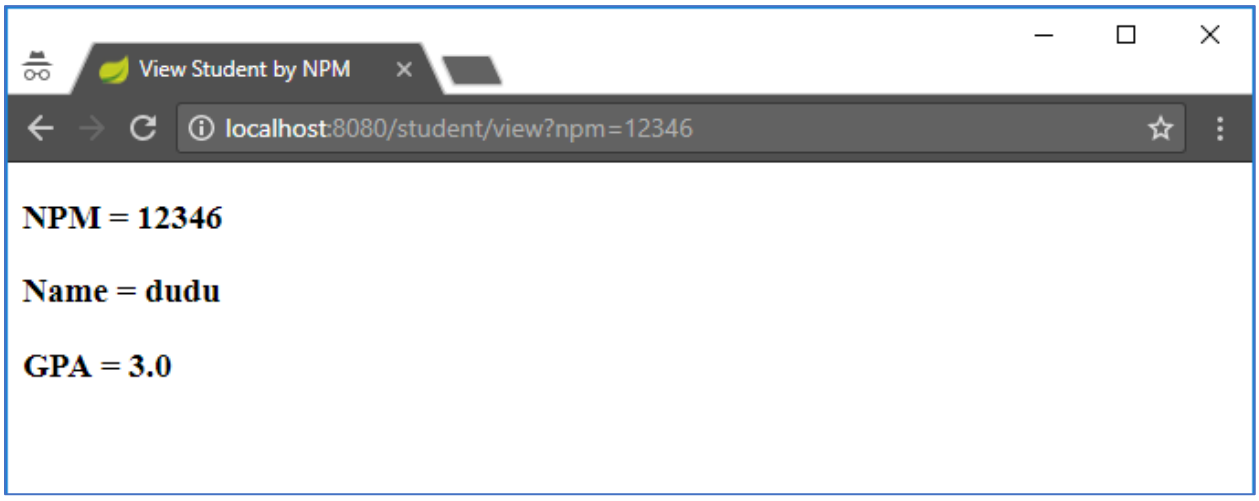
Pertanyaan 4 : Apabila program direstart dan url berikut dibuka :

```
localhost:8080/student/view?npm=12345
```

Hasilnya : page akan menampilkan error notification karena tidak ada parameter npm yang ditemukan pada Student.



Apabila data Student ditambah dengan NPM yang berbeda, maka hasil yang muncul adalah sebagai berikut :



Method View All

Pada class StudentController ditambahkan method

```
@RequestMapping("/student/viewall")
public String viewAll(Model model) {
	List<StudentModel> students = studentService.selectAllStudents();
	model.addAttribute("students", students);

	return "viewall";
}
```

Pada resources/templates >viewall.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>View All Students</title>
</head>
<body>
	<div th:each="student,iterationStatus: ${students}">
		<h3 th:text="'No ' + ${iterationStatus.count}">No. 1</h3>
		<h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
		<h3 th:text="'Name ' + ${student.name}">Student Name</h3>
		<h3 th:text="'GPA ' + ${student.gpa}">Student GPA</h3>
		<hr/>
	</div>
</body>
</html>
```

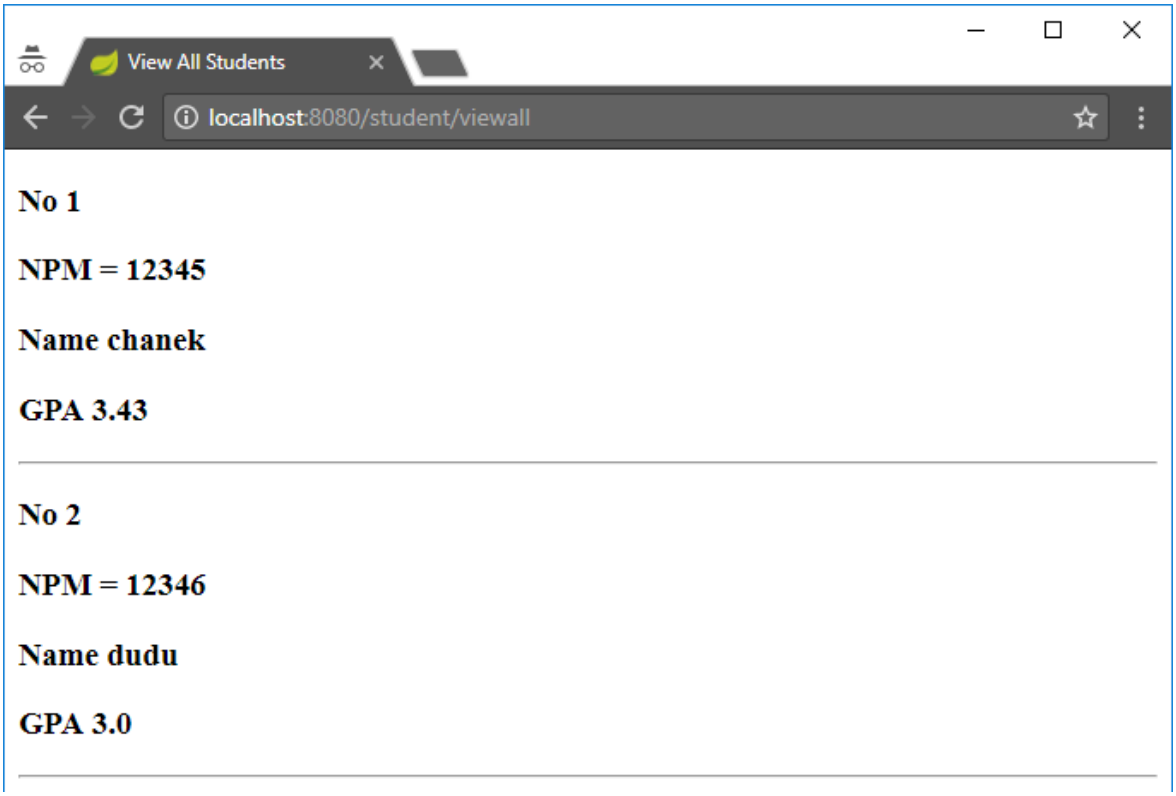
Pertanyaan 5 : Apabila program diexecute dan url berikut dibuka :

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

Lalu dibuka

```
localhost:8080/student/viewall
```

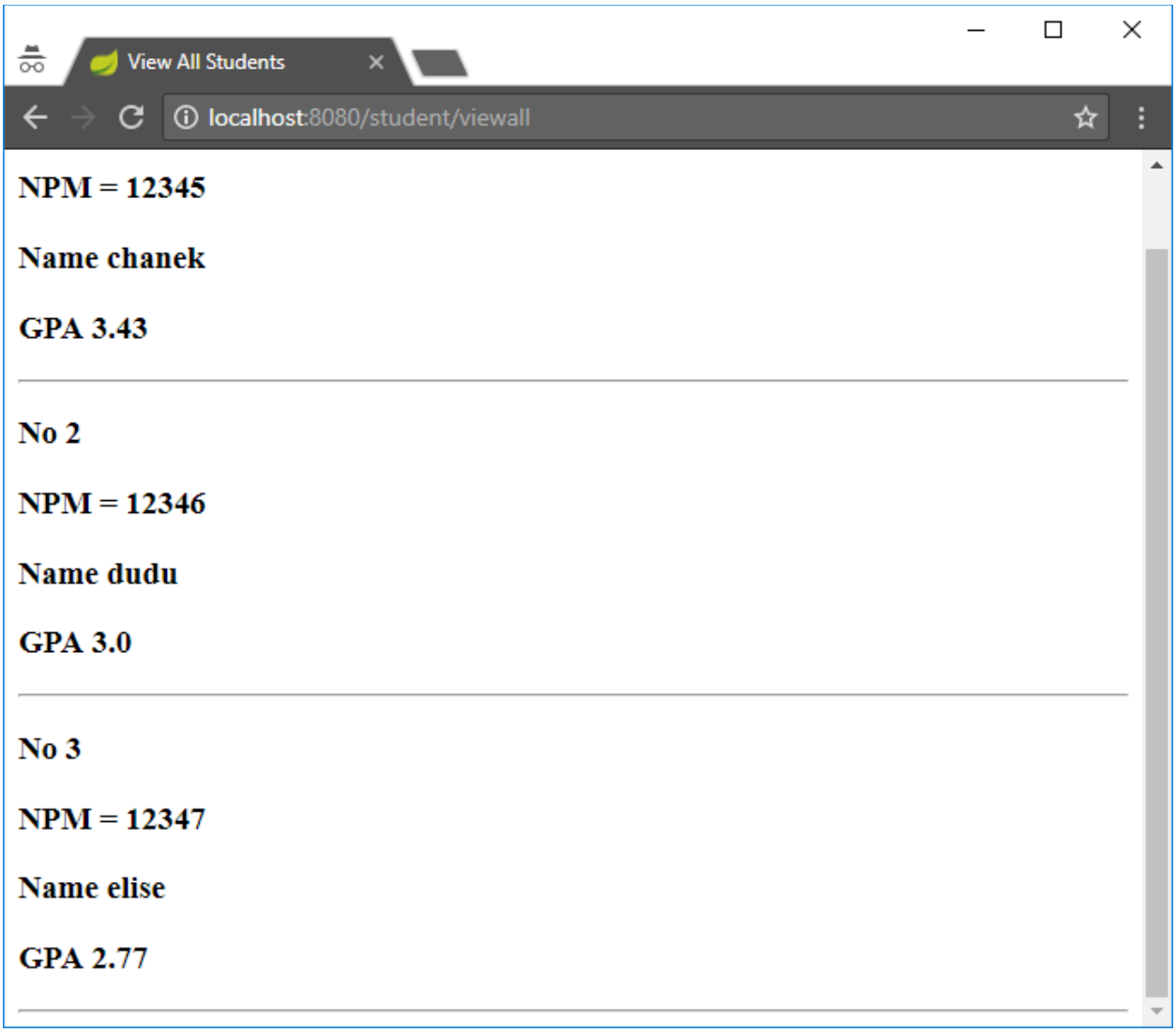
Hasilnya :



Pertanyaan 6 : Apabila data Student lainnya dengan NPM yang berbeda, dan url berikut dibuka :

localhost:8080/student/viewall

Maka hasilnya : semua record yang diinput ditampilkan di halaman tersebut.



Nama : Lidia Natalia Manik
NPM : 1706106803

Latihan

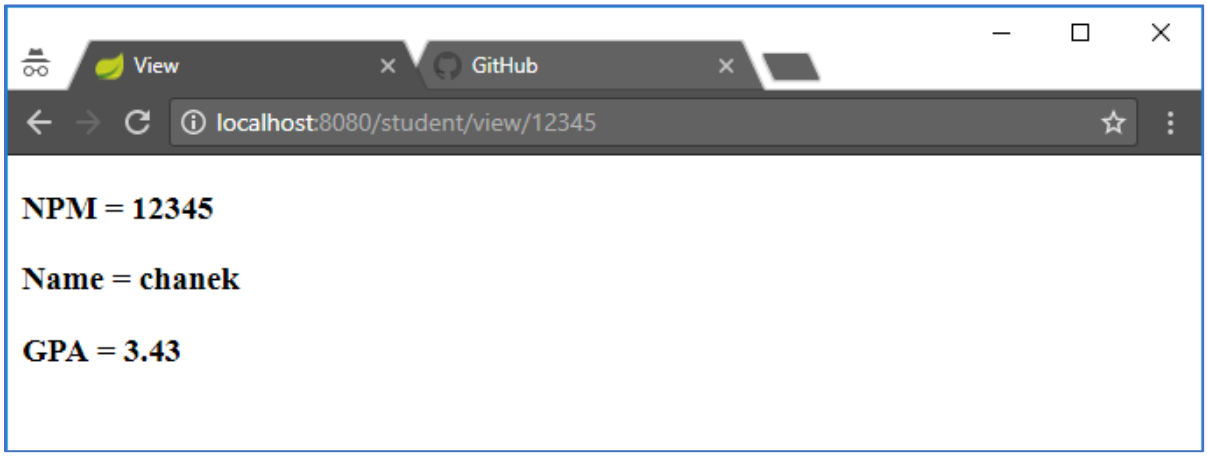
1. Menambahkan method view Student dengan menggunakan path Variable.

```
	@RequestMapping(value = {"/student/view/", "student/view/{npm}"})
	public String viewStudent (@PathVariable Optional<String> npm, Model
model) {
		if(!npm.isPresent()) {
			return "NPM kosong";
		} else {
			StudentModel student =
studentService.selectStudent(npm.get());
			if(student != null) {
				model.addAttribute("student", student);
			} else {
				return "NPM tidak ditemukan";
			}
		}
		return "view";
	}
```

Jika alamat url berikut diberikan nomor npm dan sesuai dengan daftar NPM yang tersedia.

```
localhost:8080/student/view/12345
```

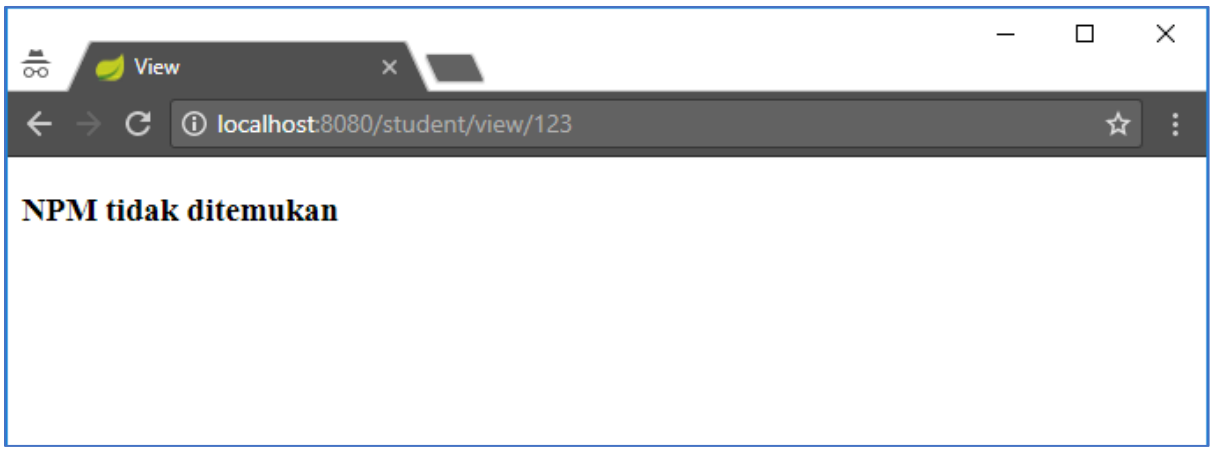
Hasilnya sebagai berikut :



Jika nomor NPM tidak diberikan pada url seperti berikut :

```
localhost:8080/student/view/123
```

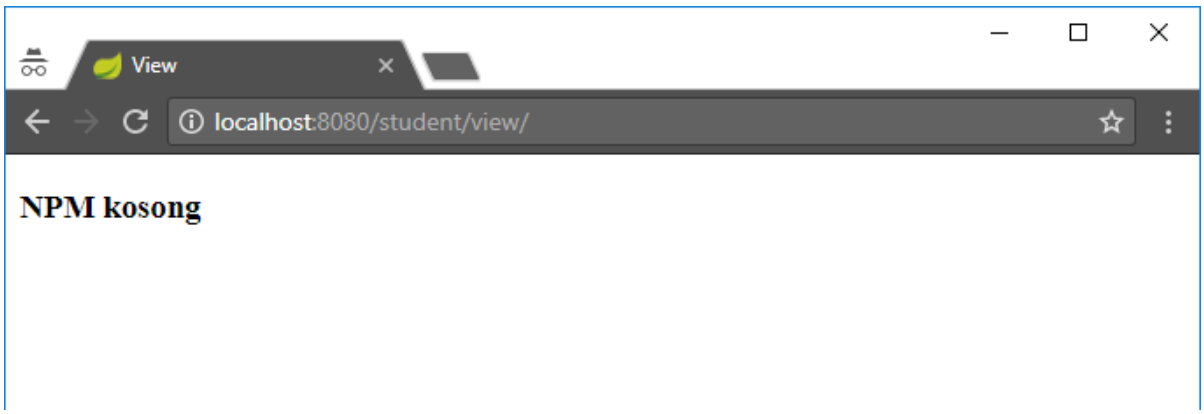
Hasilnya adalah sebagai berikut :



Jika nomor NPM diisi namun tidak ditemukan :

```
localhost:8080/student/view/12346
```

Hasilnya adalah sebagai berikut :

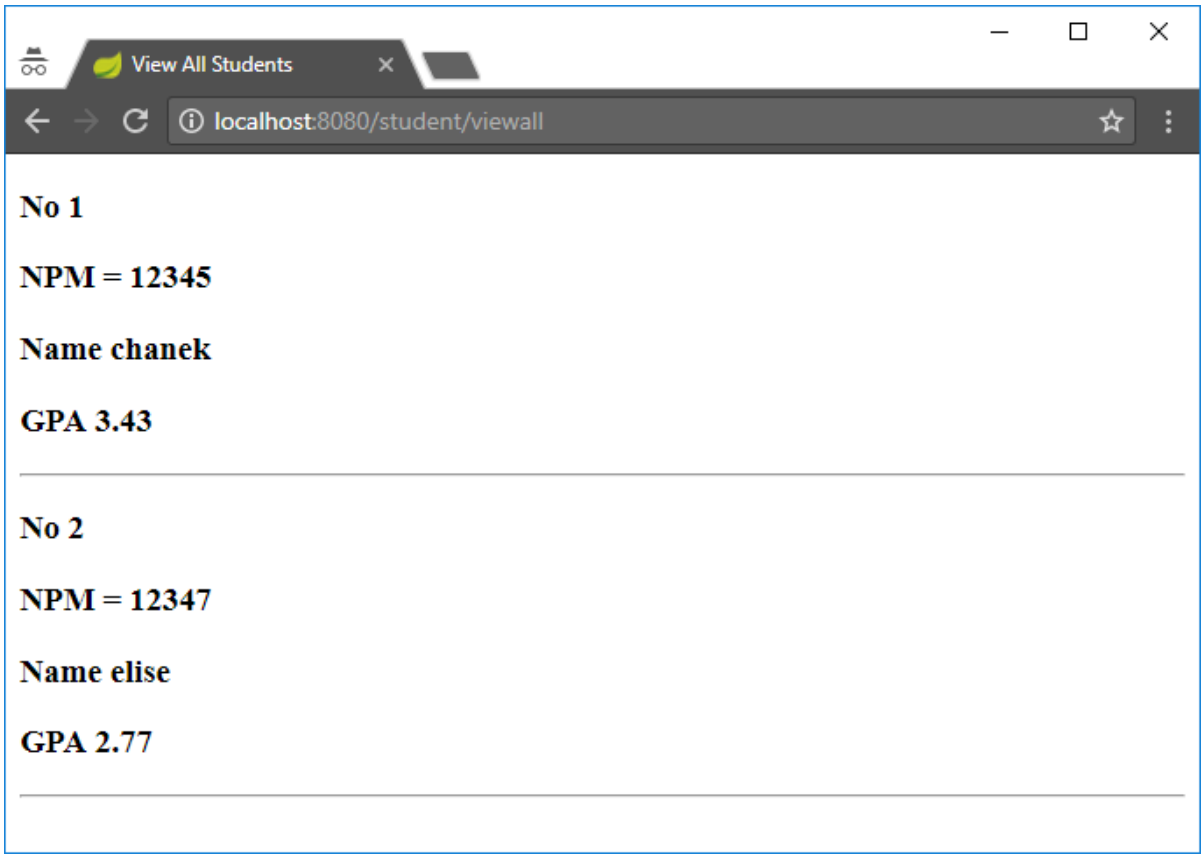


2. Menambahkan fitur delete Student berdasarkan NPM

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>Delete</title>
</head>
<body>
	<h3 th:text="${result}">Student NPM</h3>
</body>
</html>
```

```
	@RequestMapping(value = {"/student/delete", "student/delete/{npm}"})
	public String deleteStudent(@PathVariable Optional<String> npm, Model
model) {
		if(!npm.isPresent()) {
			model.addAttribute("result","NPM kosong");
		} else {
			Boolean flag = studentService.deleteStudent(npm.get());
			if(flag) {
				model.addAttribute("result","Berhasil menghapus NPM
: "+npm.get());
			} else {
				model.addAttribute ("result","NPM kosong atau tidak
ditemukan. Proses delete dibatalkan.");
			}
		}
		return "delete";
	}
```

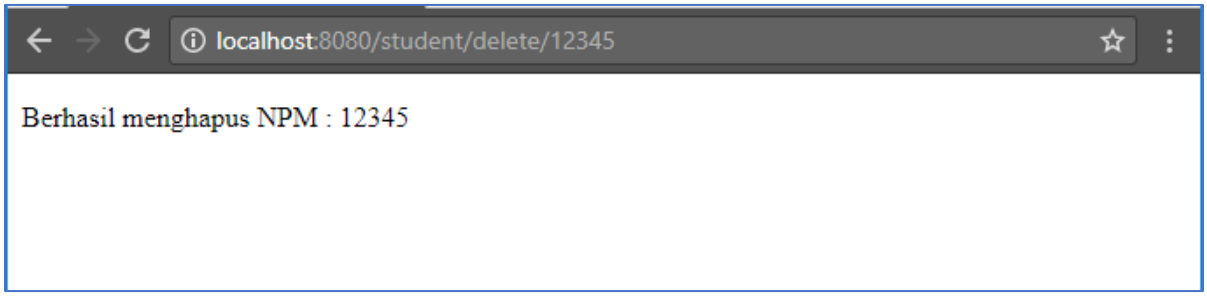
Pada tampilan awal, data yang tersedia adalah sebagai berikut :



Jika alamat url berikut diberikan nomor NPM dan sesuai dengan daftar NPM yang tersedia:

```
localhost:8080/student/view/12345
```

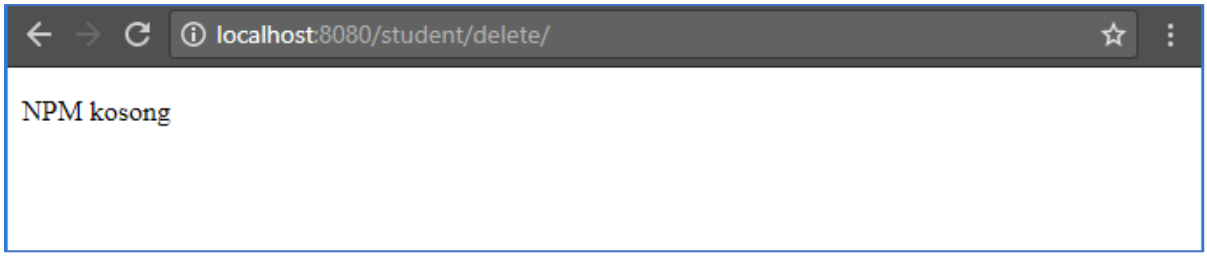
Hasilnya sebagai berikut :



Jika nomor NPM tidak diberikan pada url seperti berikut :

localhost:8080/student/delete

Hasilnya adalah sebagai berikut :



Jika nomor NPM diisi namun tidak ditemukan :

localhost:8080/student/delete/12

Hasilnya adalah sebagai berikut :