[**ARSITEKTUR DAN PEMROGRAMAN APLIKASI ENTERPRISE**](https://scele.cs.ui.ac.id/course/view.php?id=449)

**“TUTORIAL 3”**

****

**Disusun oleh:**

**Rahmawati Putrianasari**

**1706106923**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS INDONESIA**

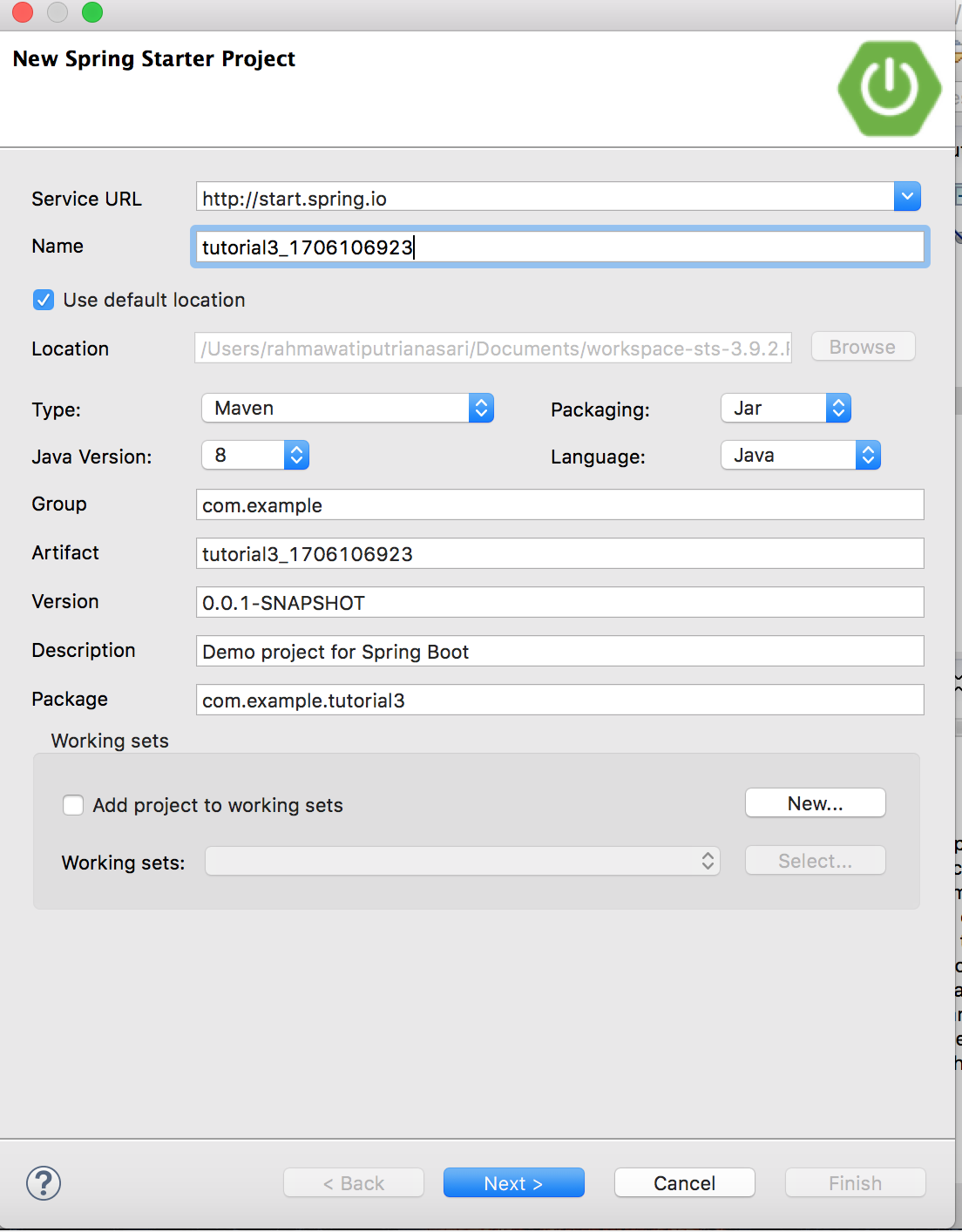
**2018**

**Tutorial**

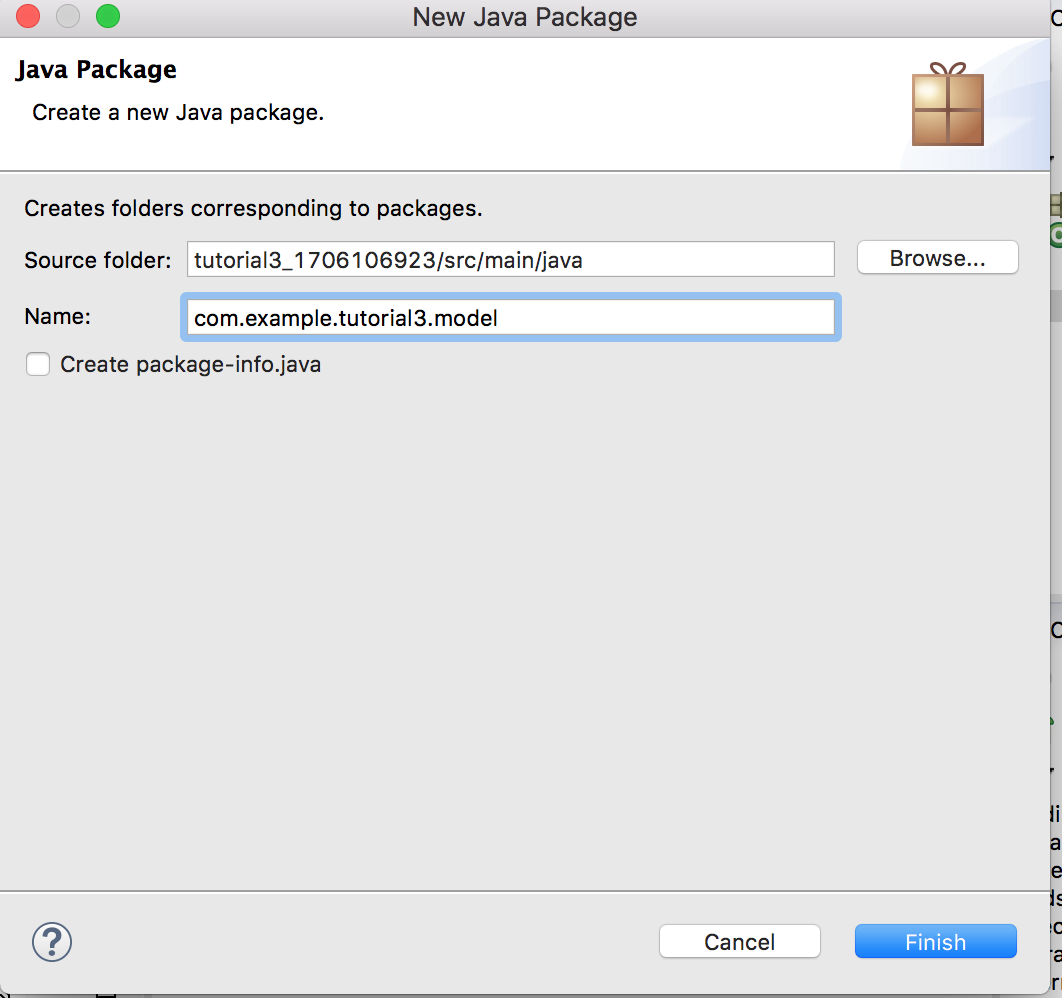
Pendahuluan

1. Pada Eclipse/STS klik File > New > Other > **Spring Boot** > Spring Starter Project, tekan Next

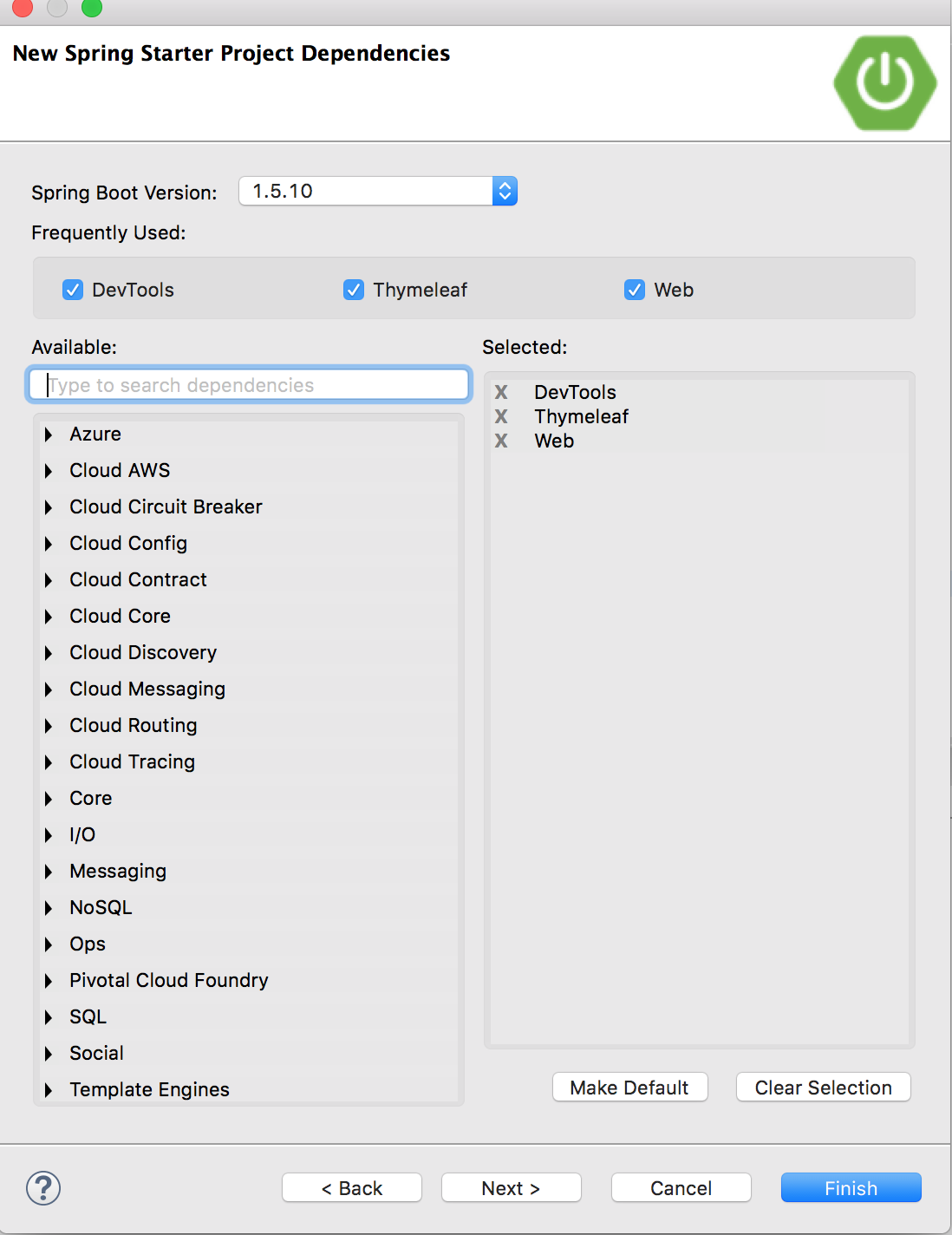
Membuat project tutorial3\_1706106923 terlebih dahulu

****

1. Beri nama *project* , *description* , *package,* dll. **Beri nama package com.example.tutorial3.**

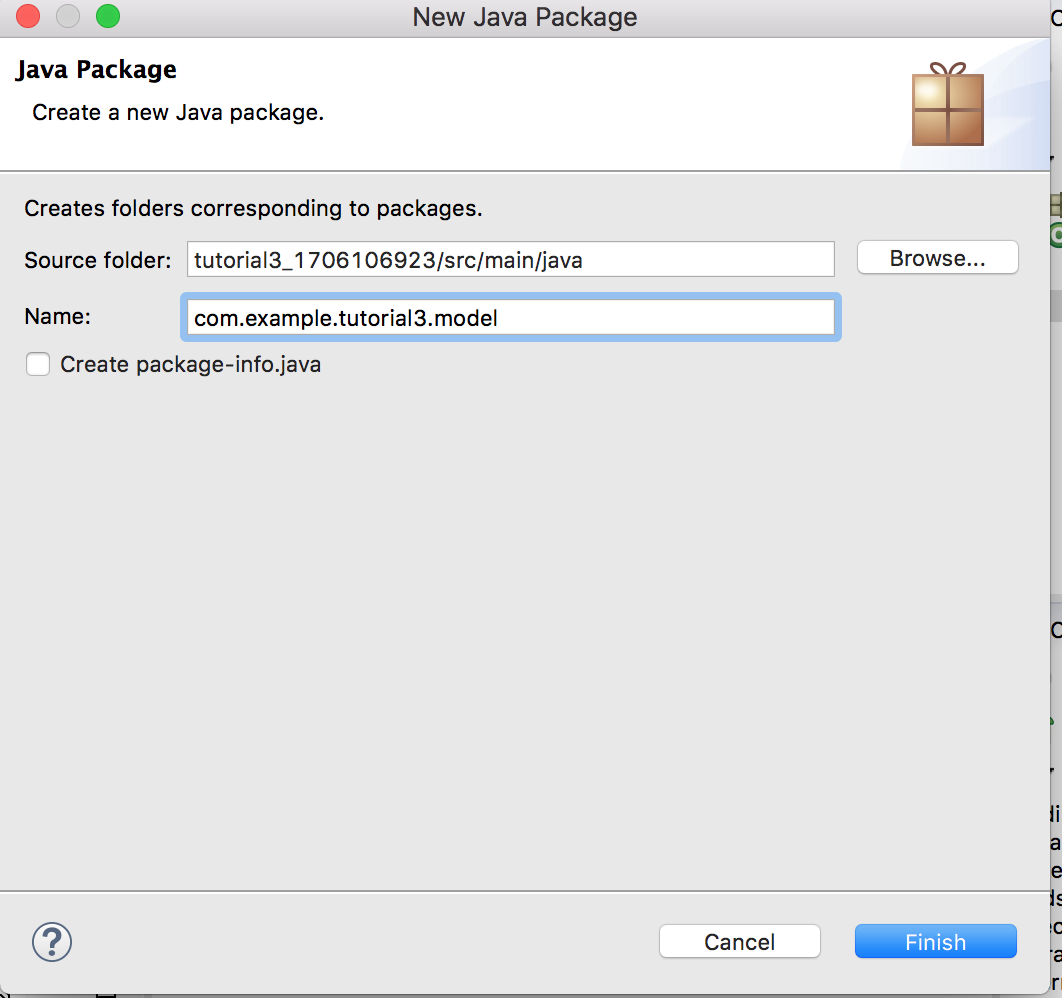
****

3. Pilih **Web** , **Thymeleaf** , dan **DevTools** pada bagian *dependency*



**Membuat *Class Model***Salah satu komponen dalam MVC adalah *model* . Anda akan membuat *model* berupa *class*

Student yang berisi variabel name (String), npm (String), dan gpa (double).  
1. Klik kanan pada Project > New > Package. Buatlah package **com.example.tutorial3.model**



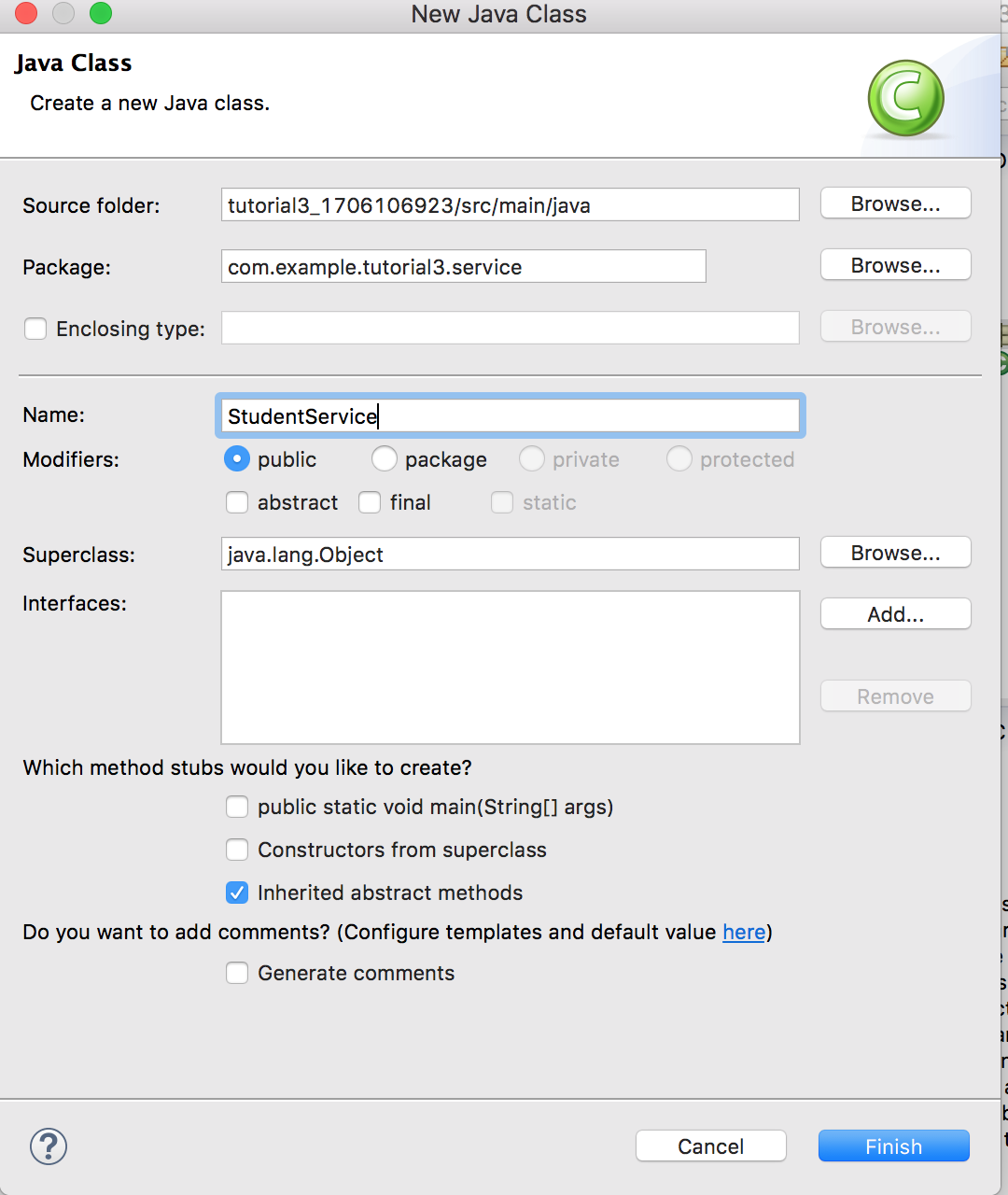
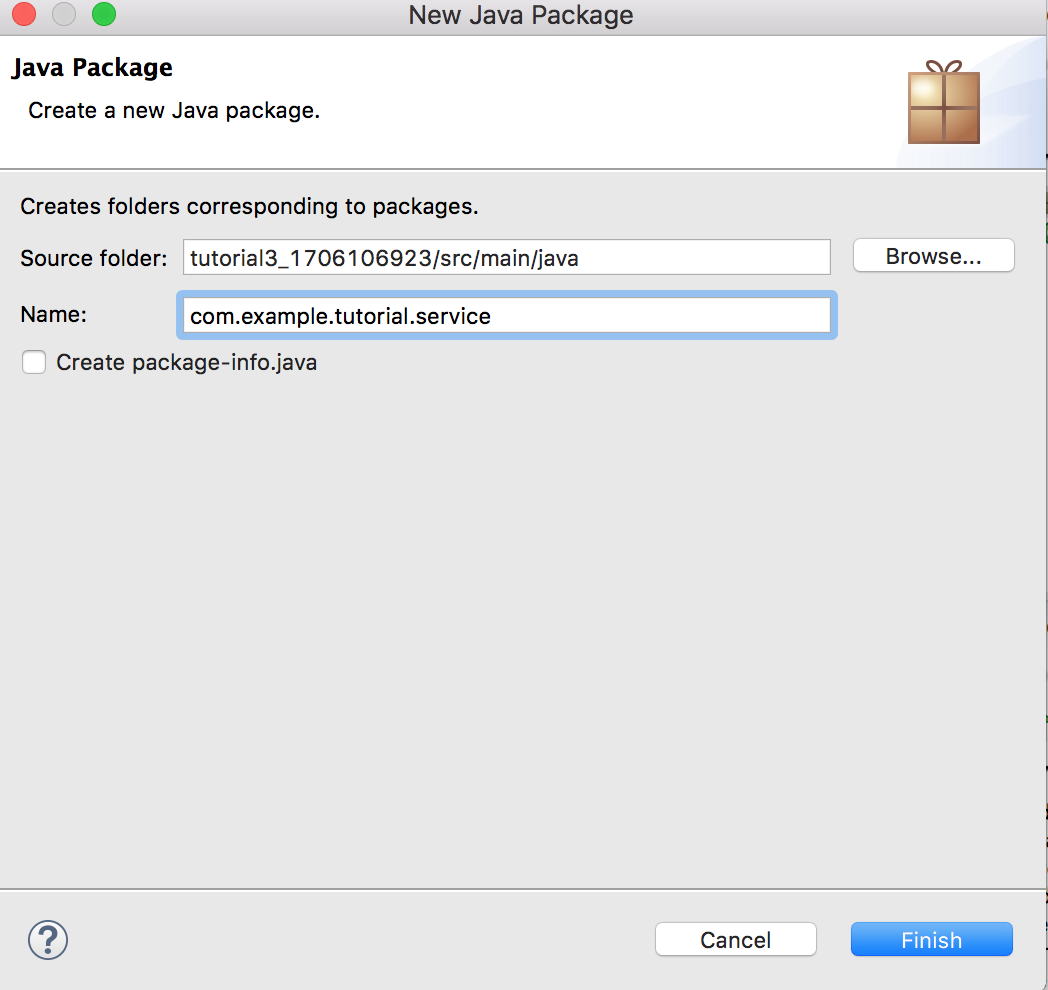
3. Tambahkan *method constructor* , *setter* , dan *getter* .

Berikut merupakan kode untuk student model dengan setter dan getternya:

|  |
| --- |
| package com.example.tutorial3.model;  public class StudentModel {  private String name;  private String npm;  private double gpa;    public StudentModel(String npm, String name, double gpa) {  this.name = name;  this.npm = npm;  this.gpa = gpa;  }    public void setName(String name) {  this.name = name;  }    public String getName() {  return name;  }  public void setNpm(String npm) {  this.npm = npm;  }    public String getNpm() {  return npm;  }    public void setGpa(double gpa) {  this.gpa = gpa;  }    public double getGpa() {  return gpa;  }    } |

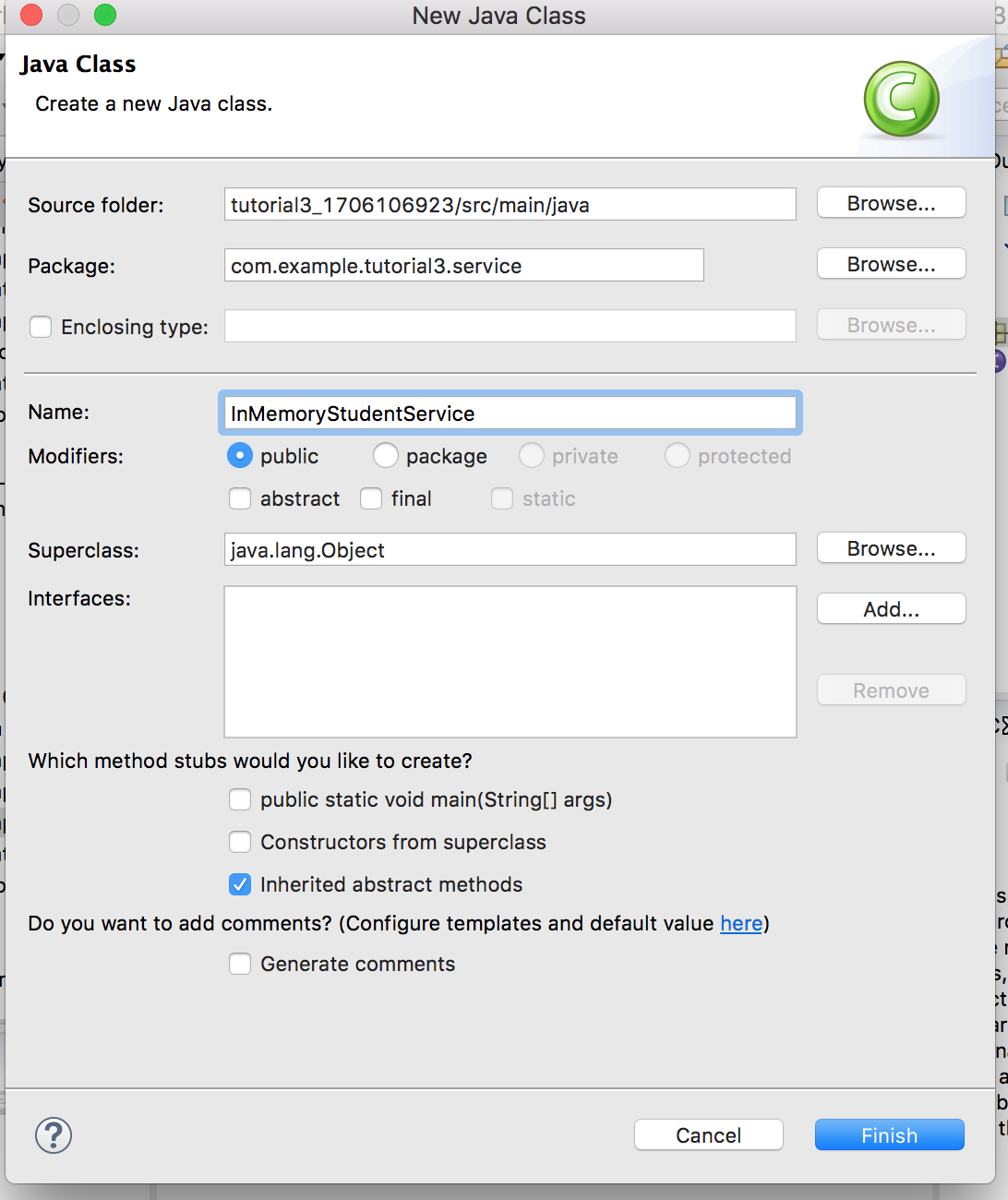
**Membuat *Service***

1. Klik kanan pada Project > New > Package. Buatlah package **com.example.tutorial3.service** 2. Buatlah *interface* **StudentService.java** pada *package* tersebut dengan isi sebagai berikut:



|  |
| --- |
| package com.example.tutorial3.service;  import java.util.List;  import com.example.tutorial3.model.StudentModel;  public interface StudentService {  StudentModel selectStudent(String npm);    List<StudentModel> selectAllStudents();    void addStudent(StudentModel student);    } |

3. Pada package yang sama, buat class InMemoryStudentService yang meng- *implements* StudentService dengan isi sebagai berikut:

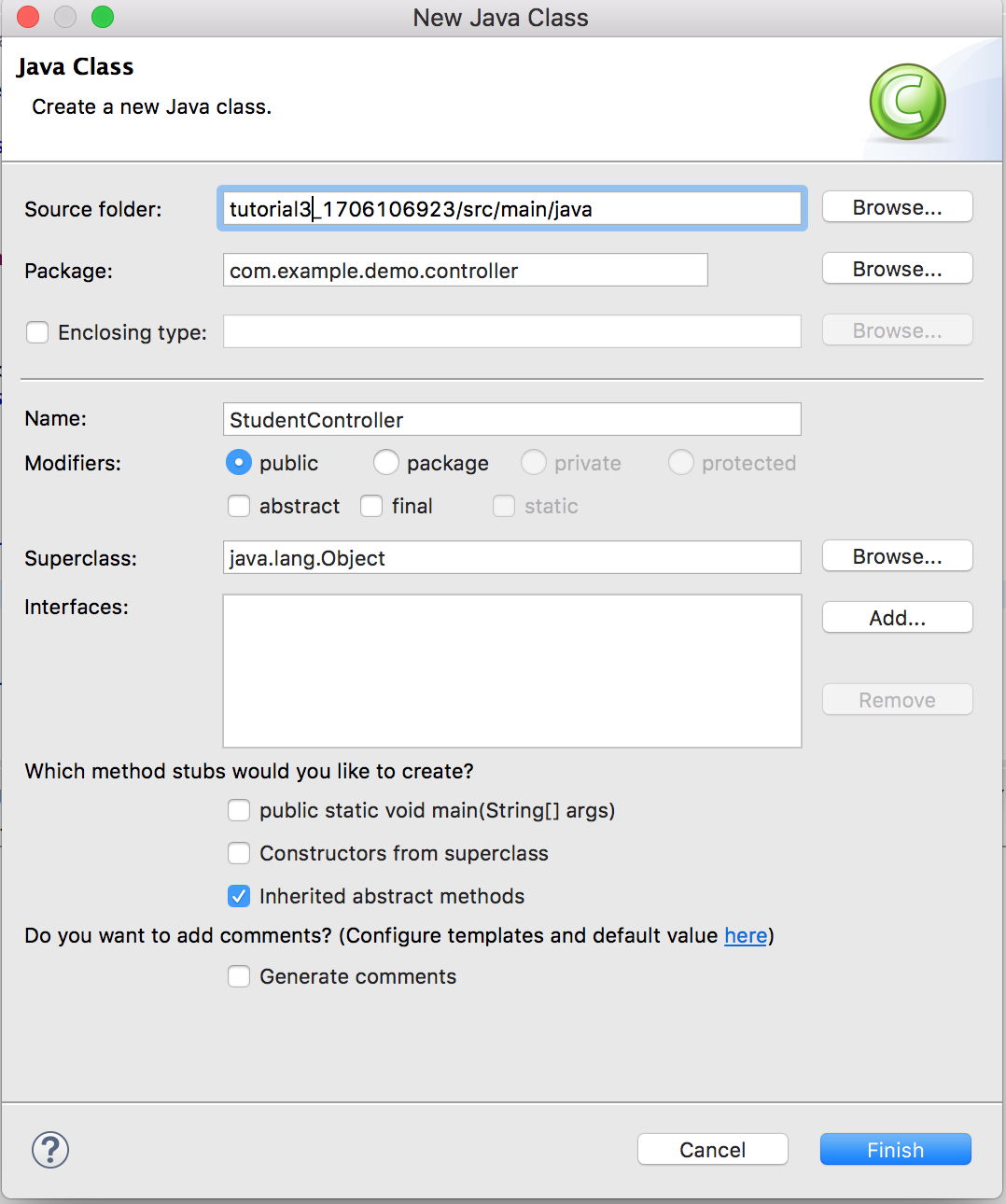
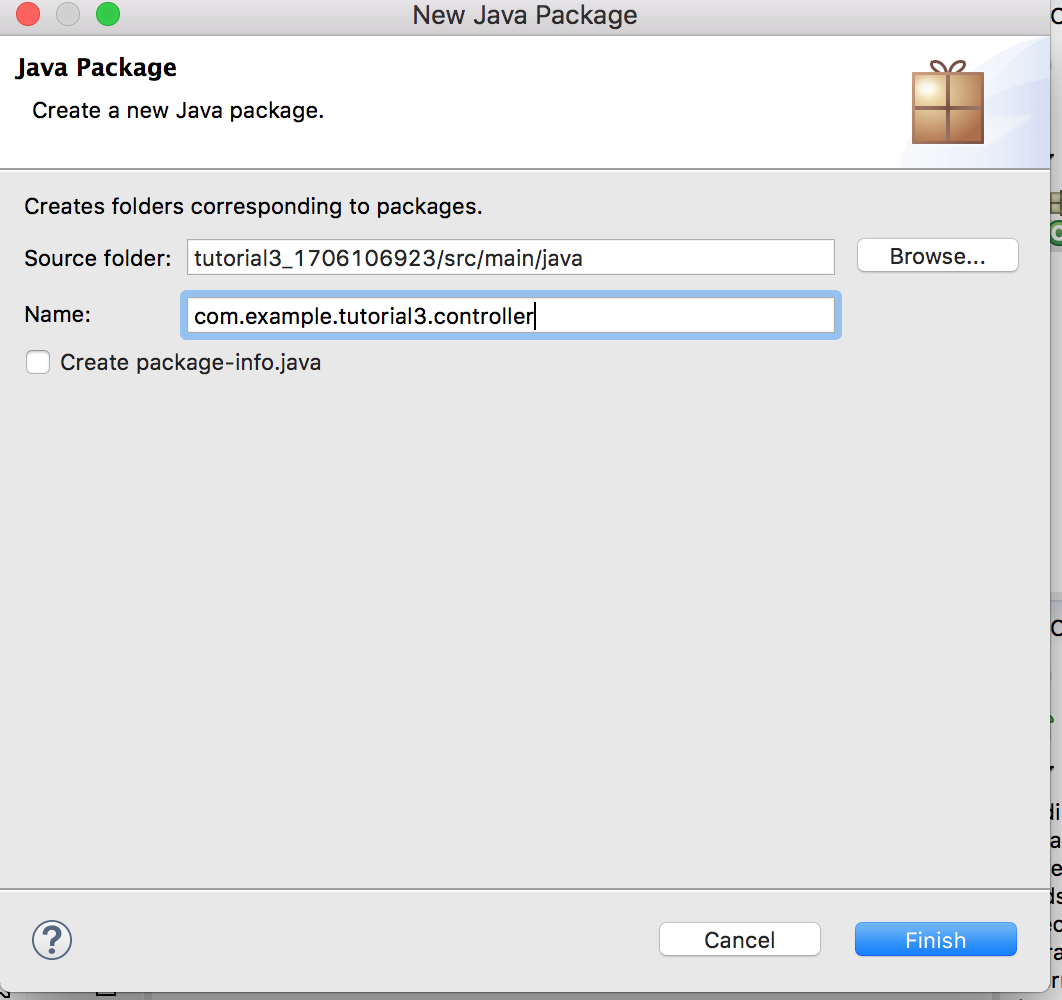


|  |
| --- |
| package com.example.tutorial3.service;  import java.util.ArrayList;  import java.util.List;  import com.example.tutorial3.model.StudentModel;  public class InMemoryStudentService implements StudentService {  private static List<StudentModel> studentList = new ArrayList<StudentModel>();    @Override  public StudentModel selectStudent(String npm) {  for(int aa = 0; aa < studentList.size(); aa++) {  if(studentList.get(aa).getNpm().equals(npm)) {  return studentList.get(aa);  }  }  return null;  }    @Override  public List<StudentModel> selectAllStudents() {  return studentList;  }    @Override  public void addStudent(StudentModel student) {  studentList.add(student);  }  } |

4. Implementasikan *method* selectStudent! *Method* ini menerima NPM mahasiswa dan mengembalikan *object* Student dengan NPM tersebut. Return null jika tidak ditemukan.

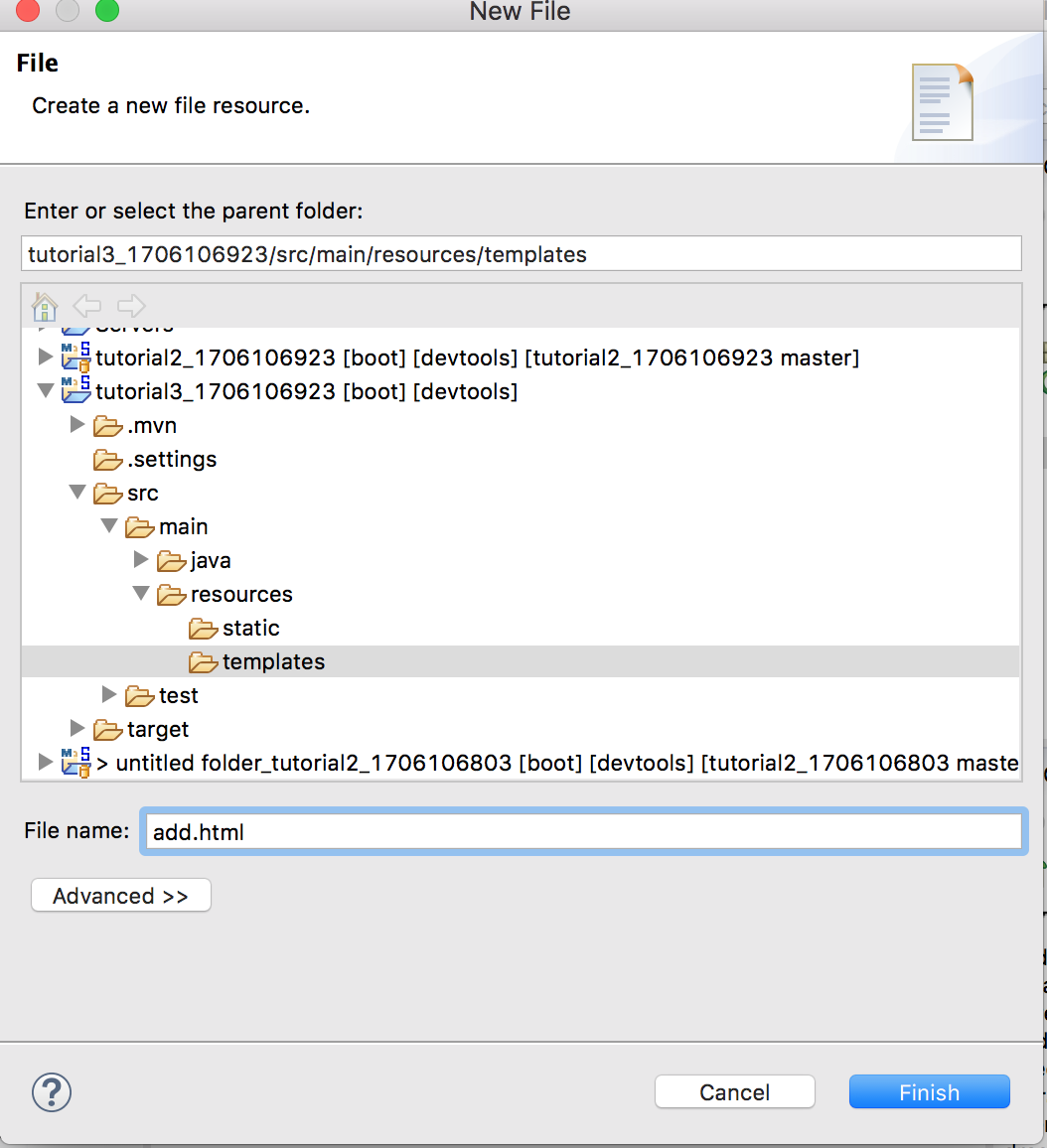
1. Klik kanan pada Project > New > Package. Buatlah *package* **com.example.tutorial3.controller**

2. Tambahkan method *add* yang menerima parameter name, npm, dan gpa dengan menggunakan request method GET. Buatlah *class* **StudentController** dengan isi sebagai berikut:



|  |
| --- |
| 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;  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";    }  } |

3. Selanjutnya pada *directory* resources/templates tambahkan add.html dengan isi sebagai berikut:

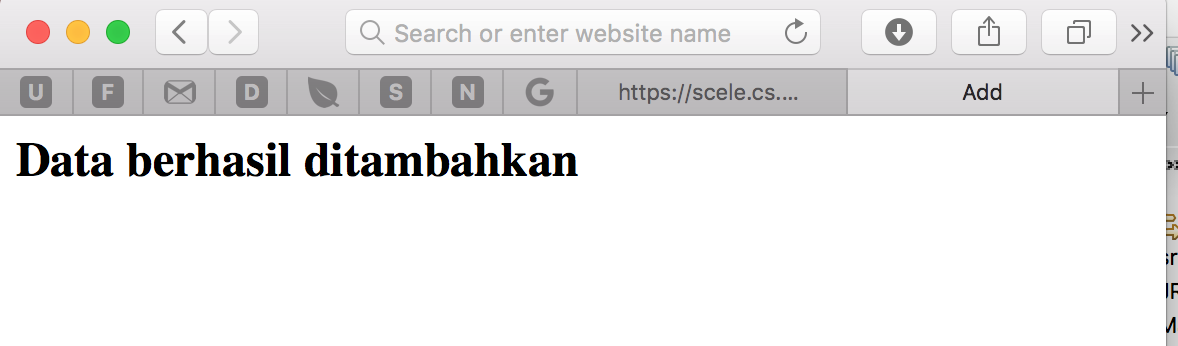


|  |
| --- |
| <html>  <head>  <title>Add</title>  </head>  <body>  <h2>Data berhasil ditambahkan</h2>  </body>  </html> |

1. Jalankan program dan buka localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43

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

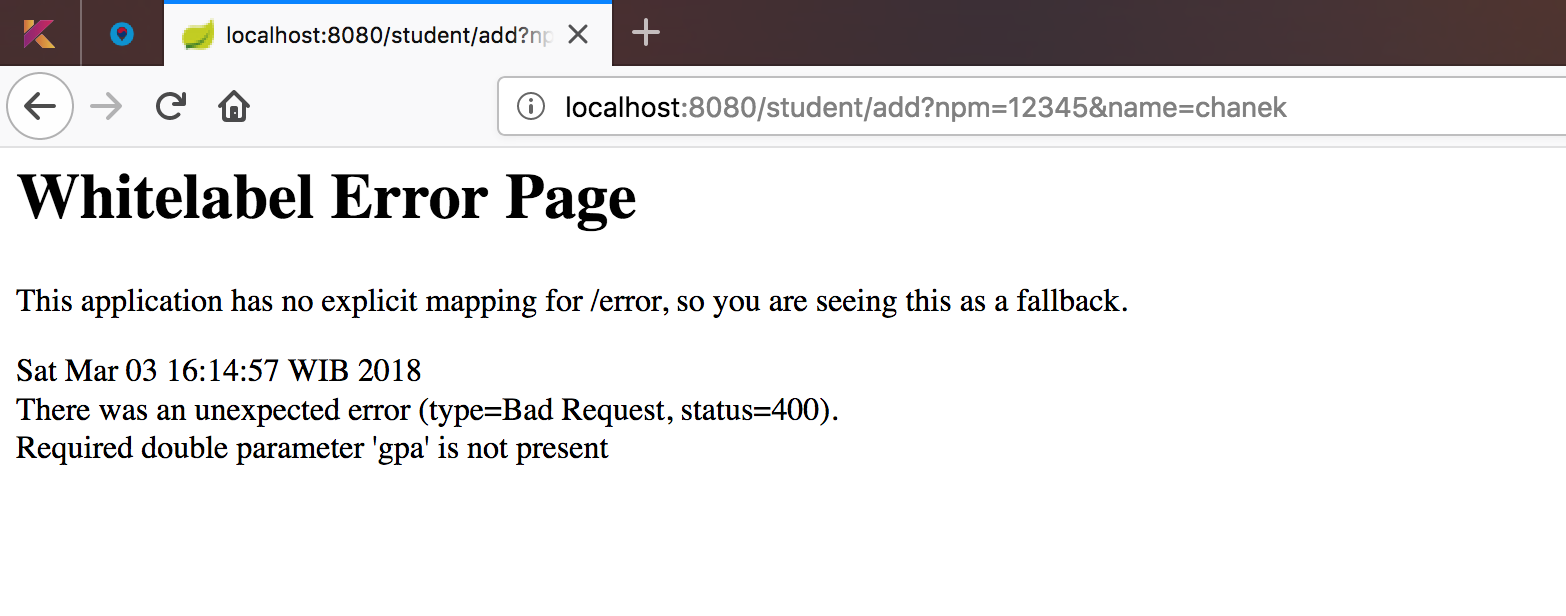
**Jawab: Dari link tersebut untuk menambahkan mahasiswa dengan mengisi npm, name, gpa dan data tersebut berhasil dimasukkan karena parameter yang di berikan benar dan datanya tidak null**

****

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

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

**Jawab: terjadi eror karena parameter yang diberikan tidak lengkap padahal parameter harus diisi karena required.**



**Method View by NPM**

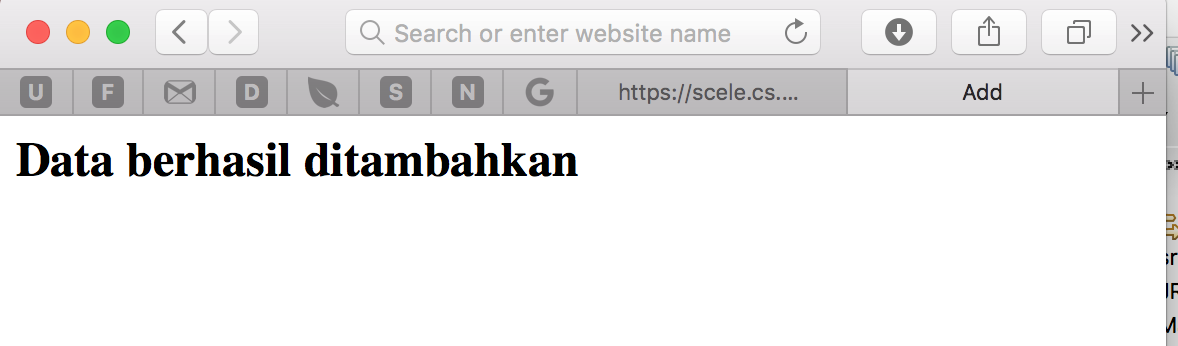
1. Pada class StudentController tambahkan *method* berikut:

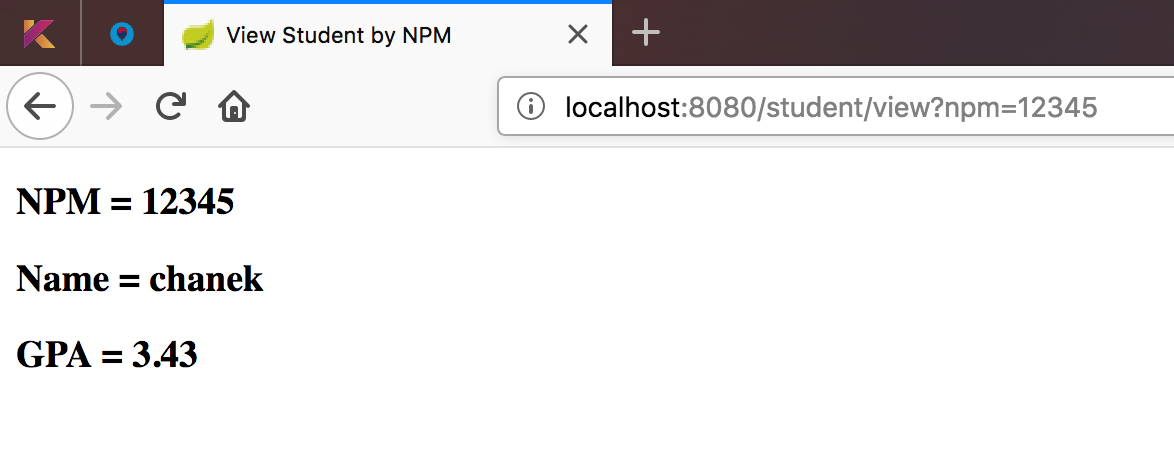
|  |
| --- |
| package com.example.tutorial3.controller;  import org.springframework.stereotype.Controller;  import org.springframework.ui.Model;  import org.springframework.web.bind.annotation.RequestMapping;  import org.springframework.web.bind.annotation.RequestParam;  import com.example.tutorial3.service.InMemoryStudentService;  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";  }    @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";  }  } |

2. Selanjutnya pada *directory* resources/templates tambahkan view.html dengan isi sebagai berikut:

|  |
| --- |
| <!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> |

3.Jalankan program dan buka localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43 lalu buka localhost:8080/student/view?npm=12345



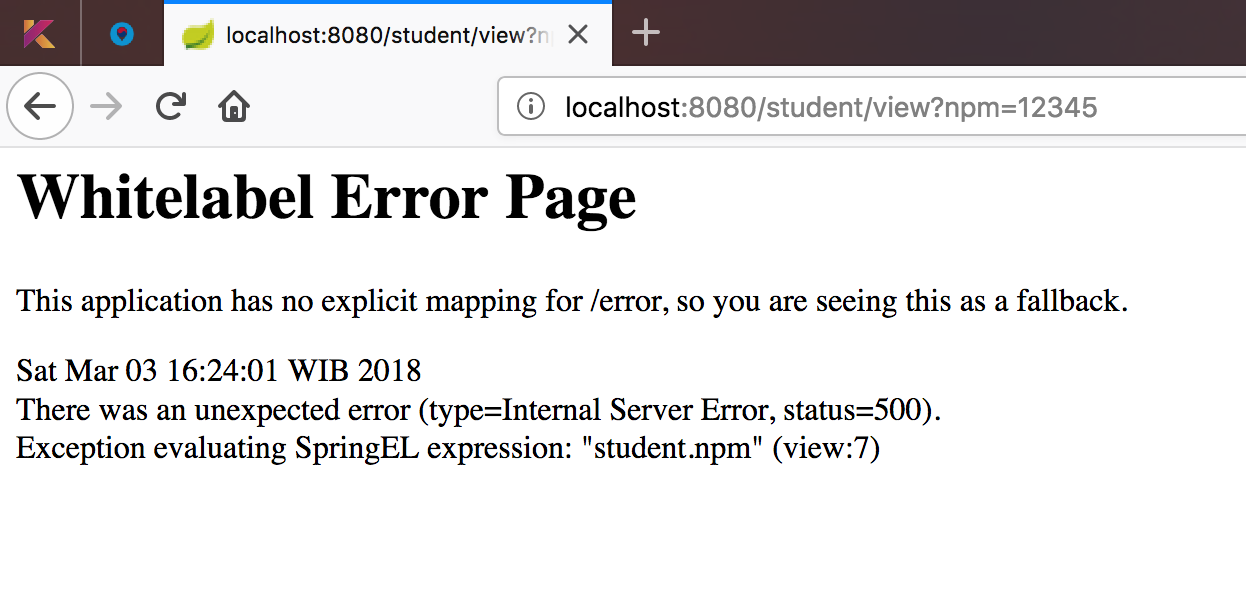


**Pertanyaan 3 : apakah data Student tersebut muncul? Jika tidak, mengapa? 4. Coba matikan program dan jalankan kembali serta buka**

**Jawab: Muncul, dikarenakan sebelumnya kita memasukkan data dengan fungsi add**

4. Coba matikan program dan jalankan kembali serta buka

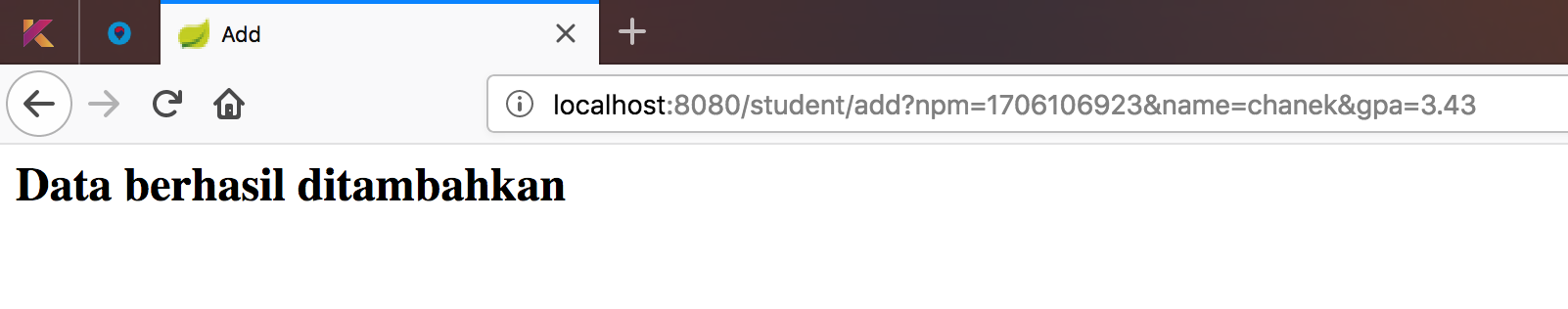
localhost:8080/student/view?npm=12345

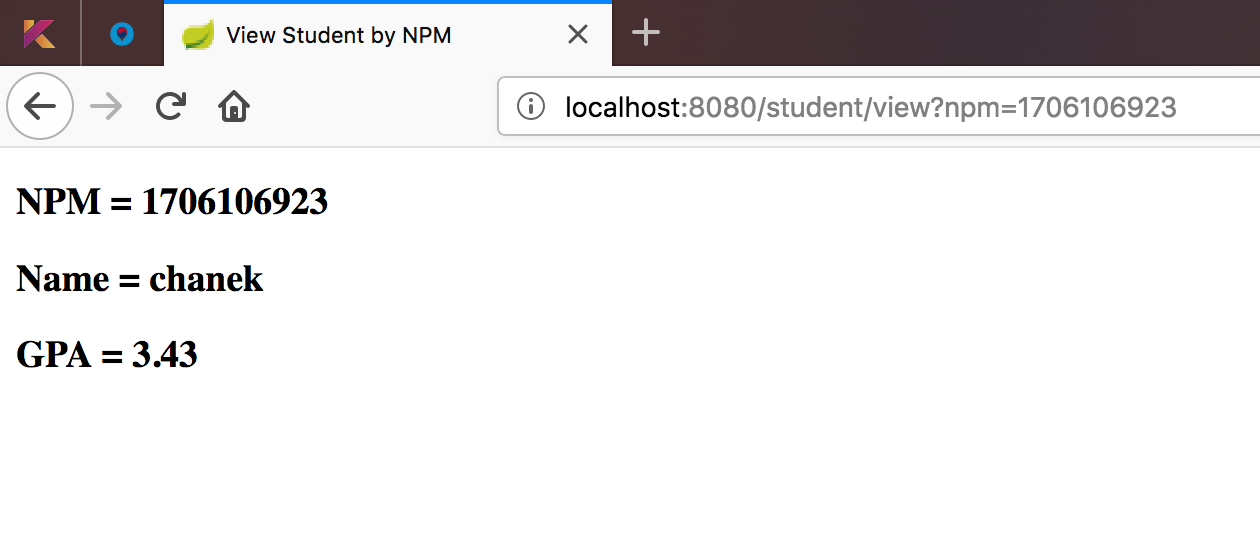


**Pertanyaan 4** : apakah data Student tersebut muncul? Jika tidak, mengapa?

**Jawab: tidak karena data sebelumnya sudah di reset saat program dimatikan maka data yang ada adalah kosong maka terjadi error seperti gambar diatas tersebut. Solusinya adalah menambahkan data lagi dan menjalankan view lagi agar terdapat data yang di tampilkan sesuai NPM yang dimasukkan**

5. Coba tambahkan data Student lainnya dengan NPM yang berbeda.





**Method View All**

1. Pada class StudentController tambahkan *method* berikut

|  |
| --- |
| package com.example.tutorial3.controller;  import org.springframework.stereotype.Controller;  import org.springframework.ui.Model;  import org.springframework.web.bind.annotation.PathVariable;  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;  import java.util.List;  import java.util.Optional;  @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";  }    @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";  }    @RequestMapping("/student/viewall")  public String viewAll(Model model) {  List<StudentModel> students = studentService.selectAllStudents();  model.addAttribute("students", students);  return "viewall";  }  } |

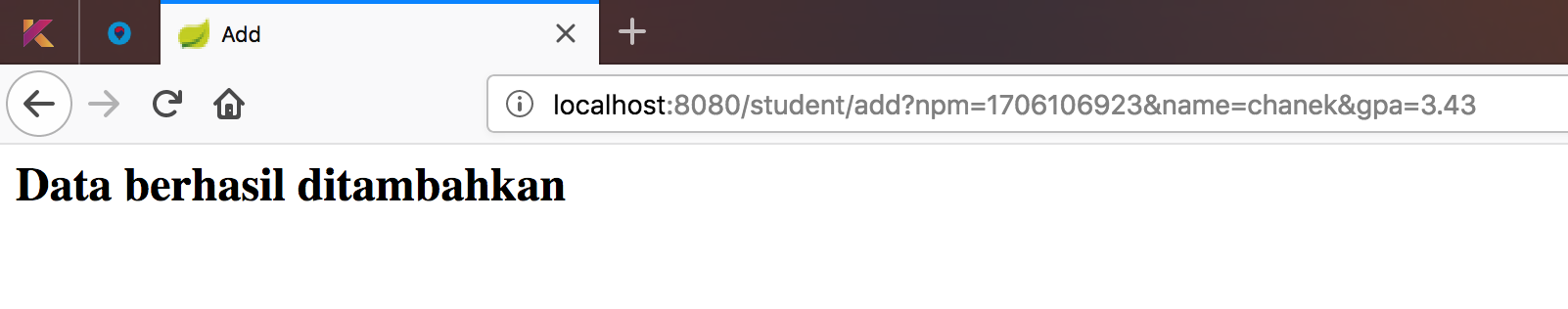
2. Selanjutnya pada *directory* resources/templates tambahkan viewall.html dengan isi sebagai berikut:

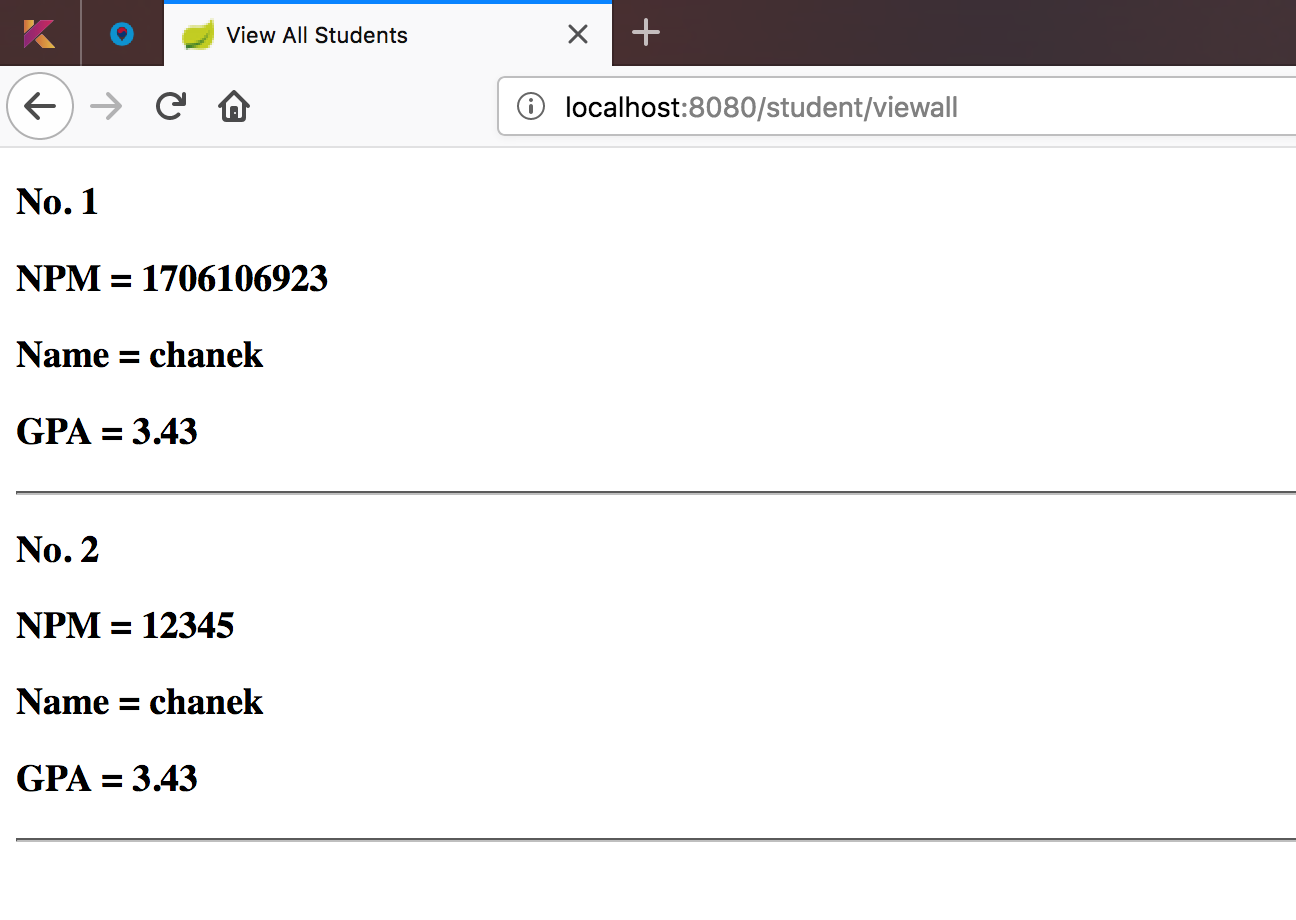
|  |
| --- |
| <!DOCTYPE html>  <html xmlns:th="http://www.thymeleaf.org">  <head>  <title>View All Students</title>  </head>  <body>  <div th:each="student,iterationStatus: ${students}">  <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>  <h3 th:text="'NPM = ' + ${student.npm}">Student NPM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

3. Jalankan program dan buka localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43 lalu buka localhost:8080/student/viewall,

4. Coba tambahkan data Student lainnya dengan NPM yang berbeda, lalu buka

localhost:8080/student/viewall,

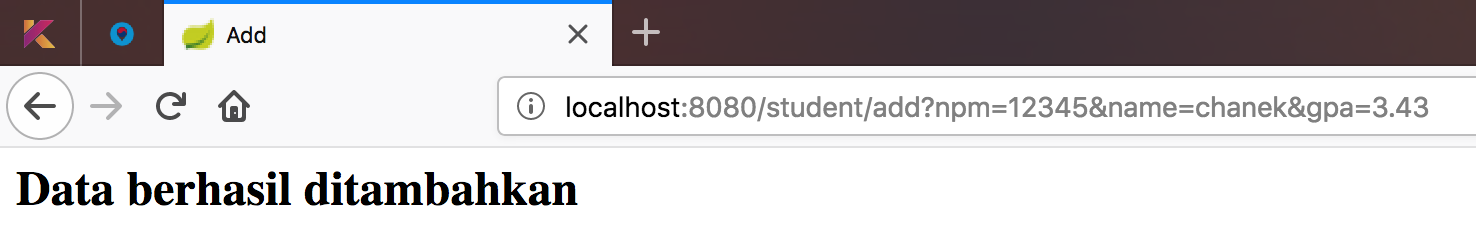


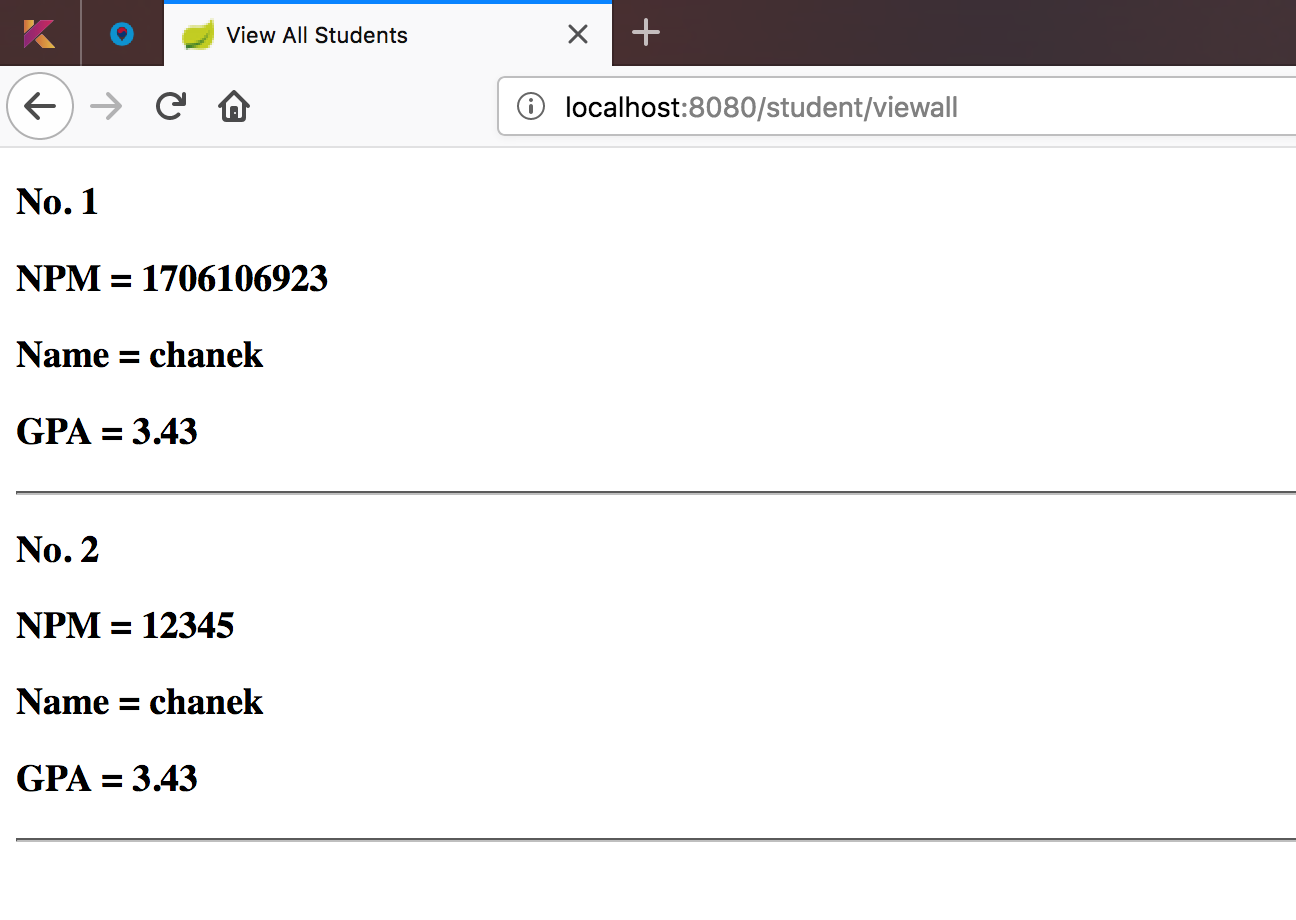


**Pertanyaan 5 : apakah data Student tersebut muncul?**

**Jawab: data muncul dan menampilkan data hanya satu karena hanya ada satu data**

**Pertanyaan 6 : Apakah semua data Student muncul?**





**Jawab: data muncul dan sekarang ada dua data dengan NPM yang berbeda**

**Latihan**

1. Pada **StudentController** tambahkan sebuah *method view* Student dengan menggunakan **Path Variable** . Misalnya, Anda ingin memasukkan data seorang Student yang memiliki NPM 14769, untuk melihat data yang baru dimasukkan tersebut dapat mengakses halaman **localhost:8080/student/view/14769.**

Jika nomor NPM tidak diberikan atau tidak ditemukan kembalikan halaman *error* yang berisi informasi bahwa nomor NPM kosong atau tidak ditemukan.

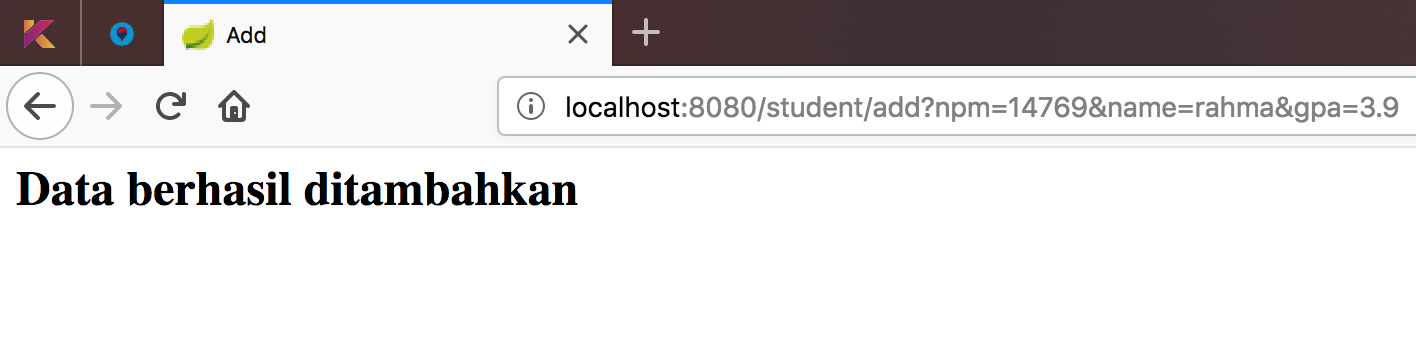
Jawab:

|  |
| --- |
| @RequestMapping(value= {"/student/view","/student/view/{npm}"})  public String studentViewPath(@PathVariable Optional<String> npm, Model model)  {  String view="";  if(npm.isPresent()) {  String Npm = npm.toString().substring(9, npm.toString().length()-1);    StudentModel student = studentService.selectStudent(Npm);  if(student == null) {  model.addAttribute("errorMessage","NPM tidak ditemukan");  view = "viewError";  }else {  model.addAttribute("student",student);  view = "view";  }  }else {  model.addAttribute("errorMessage","NPM tidak diberikan");  view = "viewError";  }  return view;  } |

PENJELASAN CODE VIEW:

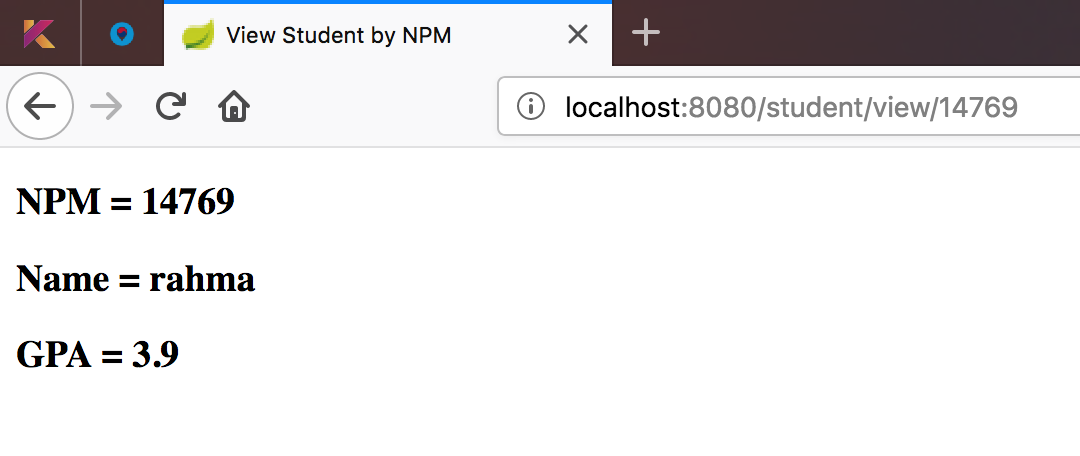
Menggunakan PathVariabel dengan npm betipe data Optional<String> lalu pertamakali di buat pengecekan apabila npm tidak diberikan maka akan menampilkan “NPM tidak diberikan”, lalu dari tipe data itu dilakukan substring untuk memotong optional<string> untuk mengambil angka NPM nya dan dilakukan pengecekan apakah data tersebut tidak ada maka menampilkan error “NPM tidak ditemukan” apabila student itu ada, maka datanya akan di tampilkan dan menggunakan view “view”. Apabila pesan error menggunakan view “viewError”

Menambahkan student dengan NPM 14769

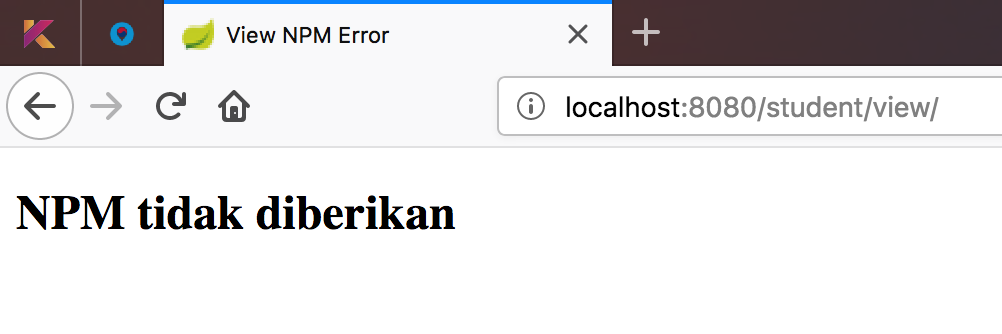


Menampilkan data:

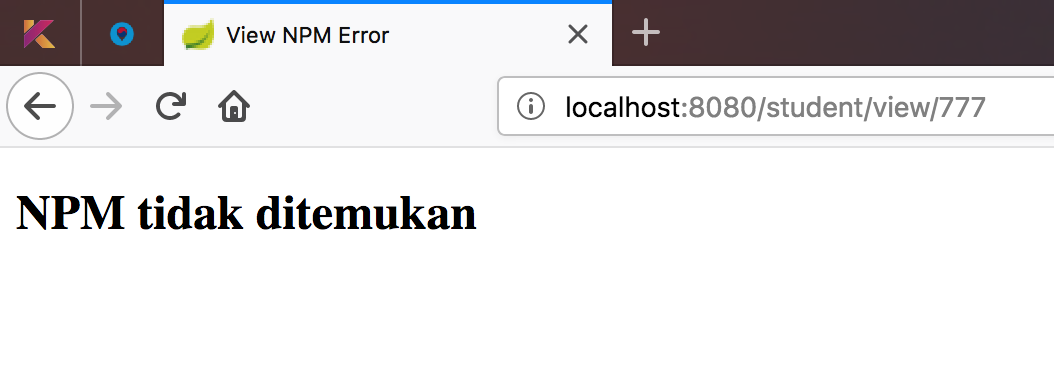
**localhost:8080/student/view/14769**



Jika NPM tidak diberikan : localhost:8080/student/view/



Jika NPM tidak terdaftar : localhost:8080/student/view/777



2. Tambahkan fitur untuk melakukan *delete* Student berdasarkan NPM. Misalnya, setelah melakukan *add* Student pada soal nomor 1, cobalah untuk melakukan *delete* data tersebut dengan mengakses halaman **localhost:8080/student/delete/14769.** Tampilkan sebuah halaman yang memberikan informasi bahwa data tersebut telah berhasil dihapus.

Jika nomor NPM tidak diberikan atau tidak ditemukan kembalikan halaman *error* yang berisi informasi bahwa nomor NPM kosong atau tidak ditemukan dan proses *delete* dibatalkan.

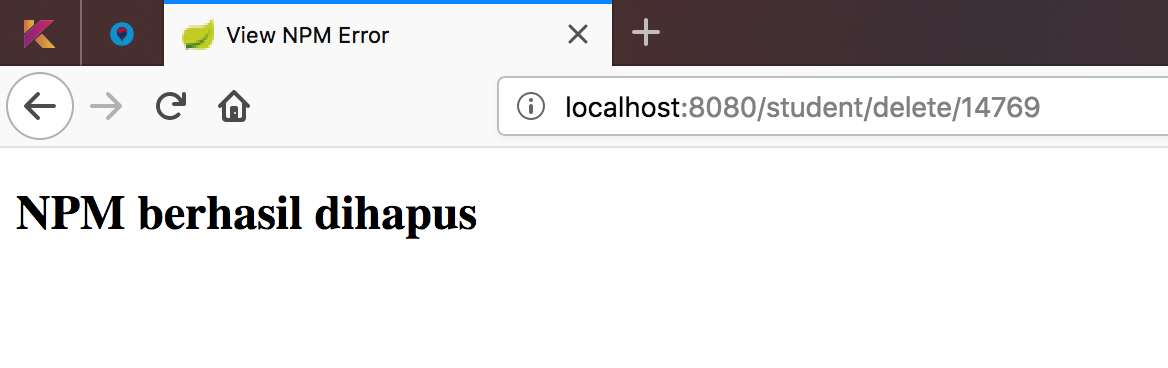
Jawab:

|  |
| --- |
| @RequestMapping(value= {"/student/delete","/student/delete/{npm}"})  public String studentDeletePath(@PathVariable Optional<String> npm, Model model)  {  if(npm.isPresent()) {  String Npm = npm.toString().substring(9, npm.toString().length()-1);    StudentModel student = studentService.selectStudent(Npm);  if(student == null) {  model.addAttribute("errorMessage","NPM tidak ditemukan");  }else {  studentService.deleteStudent(student);  model.addAttribute("errorMessage","NPM berhasil dihapus");  }  }else {  model.addAttribute("errorMessage","NPM tidak diberikan");  }  return "viewError";  } |

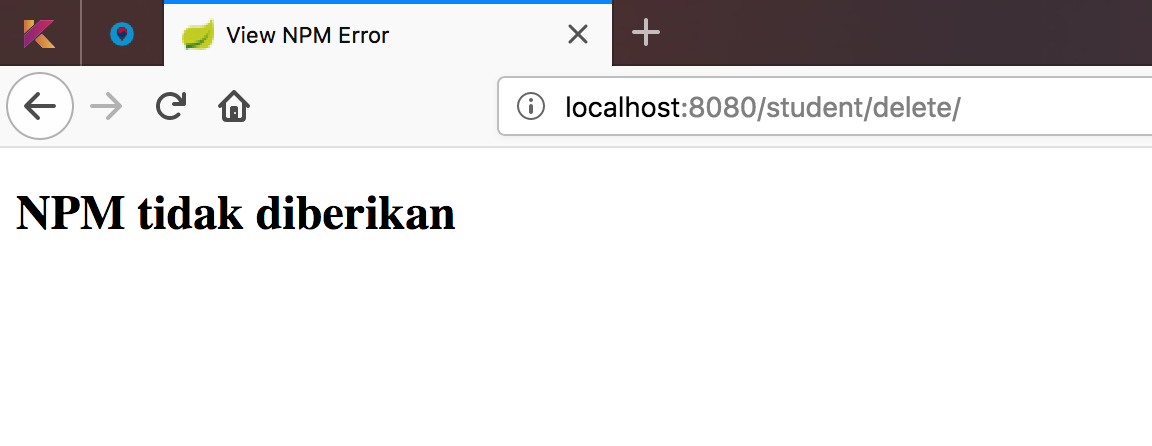
PENJELASAN CODE DELETE:

Menggunakan PathVariabel dengan npm betipe data Optional<String> dilakukan pengecekan terlebih dahulu apabila npm tidak diberikan maka akan menampilkan “NPM tidak diberikan”, lalu dari tipe data itu dilakukan substring untuk memotong optional<string> untuk mengambil angka NPM nya dan dilakukan pengecekan apakah data tersebut tidak ada maka menampilkan error “NPM tidak ditemukan” apabila student itu ada, maka datanya akan di hapus dan menampilkan view “viewError”

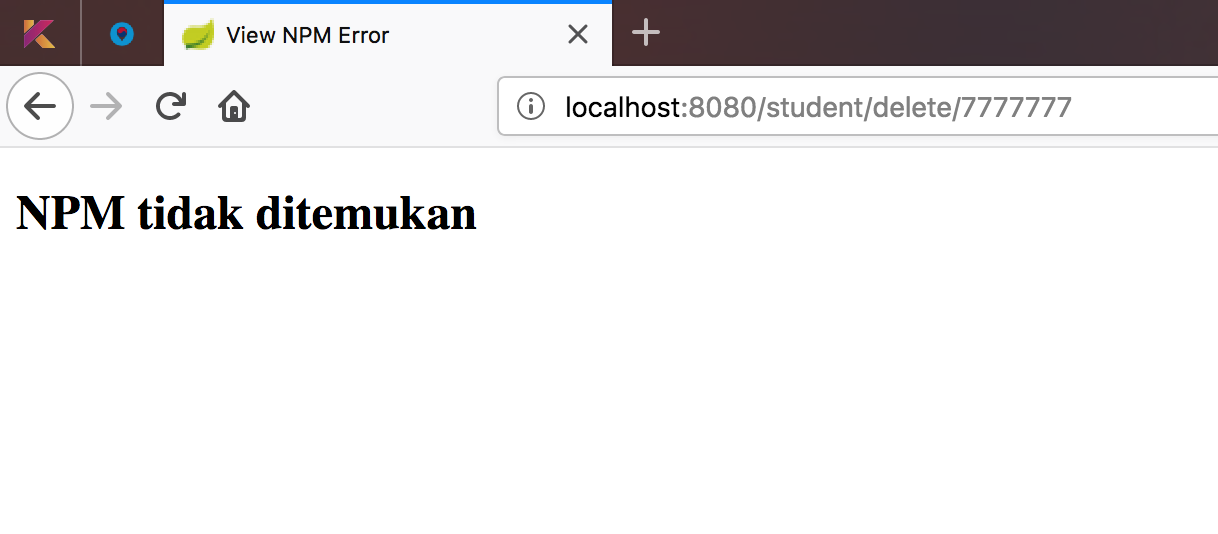
Delete data: **localhost:8080/student/delete/14769**



NPM tidak diberikan : localhost:8080/student/delete/

****

NPM tidak ditemukan : localhost:8080/student/delete/7777777



Rangkuman:

Dari tutorial ini kita belajar membuat service dengan konsep MVC mulai dengan memasukkan data dengan parameter, memasukkan data dalam sebuah list, menampilkan data berdasarkan NPM maupun menampilkan semua data (viewall) serta cara menghapus sebuah data. Dan menampilkan pesan-pesan error yang dibutuhkan.