Nama : Muhammad Kamal
NPM : 1606954905

Tutorial 04 APAP

1. Method untuk delete

- Method pada studentmapper untuk delete

```
@Delete("Delete from student where npm = #{npm}")
void deleteStudent(@Param("npm") String npm);
```

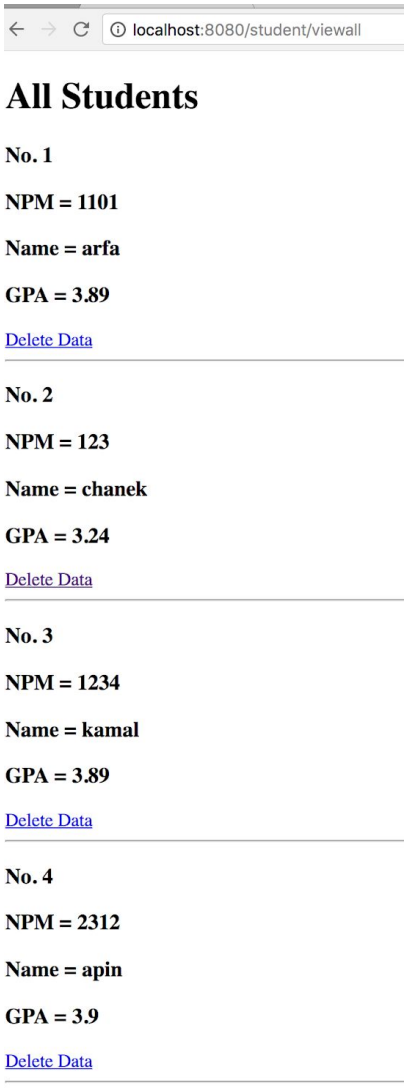
- Method delete student pada service class

```
@Override
public void deleteStudent (String npm)
{
        log.info ("student " + npm + " deleted");
        studentMapper.deleteStudent(npm);
}
```

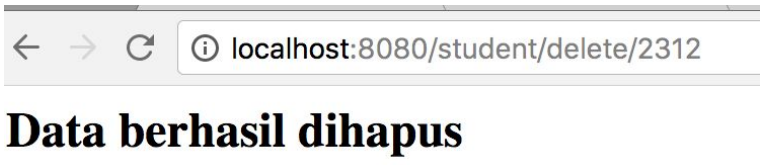
- Pada controller tambahkan pengecekan, berikut

```
@RequestMapping("/student/delete/{npm}")
public String delete (Model model, @PathVariable(value = "npm") String npm)
{
    StudentModel student = studentDAO.selectStudent (npm);

    if (student != null) {
        studentDAO.deleteStudent (npm);

        return "delete";
    } else {
        model.addAttribute ("npm", npm);
        return "not-found";
    }

}
```

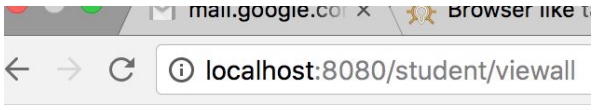
- Hasil dari tambah student, lakukan view all



- Lakukan panggilan delete, dengan klik link di view all, akan tampil sebagai berikut



Selanjutnya lakukan view all setelah delete



All Students

No. 1

NPM = 1101

Name = arfa

GPA = 3.89

Delete Data

No. 2

NPM = 123

Name = chanek

GPA = 3.24

Delete Data

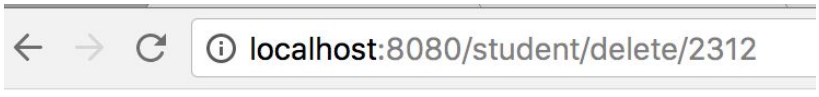
No. 3

NPM = 1234

Name = kamal

GPA = 3.89

Delete Data



Student not found

NPM = 2312

2. Method menambahkan update

- Method update pada studentmapper

```
@Update("Update student set name = #{name}, gpa = #{gpa} where npm = #{npm}")
  void updateStudent (StudentModel student);
```

- Method update student pada interface studentservice

```
void updateStudent (StudentModel student);
```

- Implementasi method update student pada studentservicedatabase

```
@Override
  public void updateStudent (StudentModel student)
  {
                log.info ("student " + student.getNpm() + " updated");
                studentMapper.updateStudent(student);
  }
```

- Tambahkan link pada view all untuk mengarahkan ke form edit untuk diupdate

```
<a th:href="'/student/update/' + ${student.npm}" > Update Data</a><br/>
```

- Tambahkan view form-update untuk melakukan edit/update

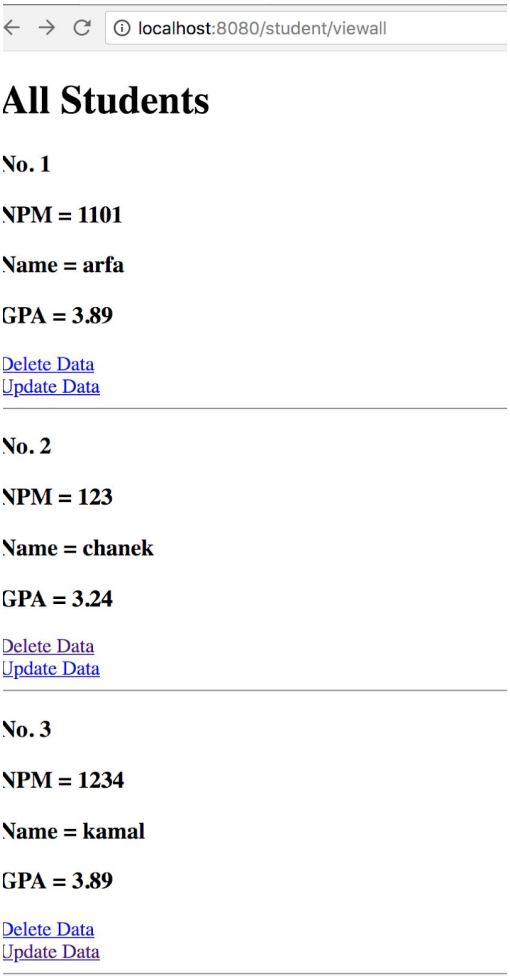
```
<!DOCTYPE HTML>
<html xmlns="http://www.w3.org/1999/xhtml"
        xmlns:th="http://www.thymeleaf.org">
<head>

<title>Update student</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>

<body>

        <h1 class="page-header">Problem Editor</h1>

        <form action="/student/update/submit" method="post">
                <div>
                        <label for="npm">NPM</label> <input type="text" readonly="true"
name="npm" th:value="${student.npm}" />
                </div>
                <div>
                        <label for="name">Name</label> <input type="text" name="name"
th:value="${student.name}" />
                </div>
                <div>
```

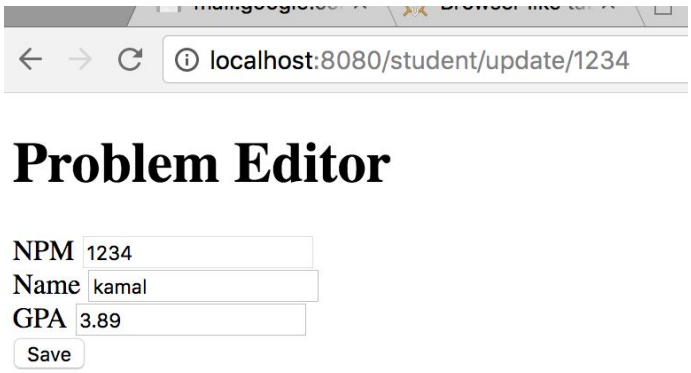
```
                        <label for="gpa">GPA</label> <input type="text" name="gpa"
th:value="${student.gpa}" />
                </div>

                <div>
                        <button type="submit" name="action" value="save">Save</button>
                </div>
        </form>

</body>

</html>
```

- Tambahkan method di studentcontroller untuk akses update student, sebagai berikut

```
@RequestMapping("/student/update/{npm}")
  public String updatePath (Model model,
        @PathVariable(value = "npm") String npm)
  {
     StudentModel student = studentDAO.selectStudent (npm);

     if (student != null) {
        model.addAttribute ("student", student);
        return "form-update";
     } else {
        model.addAttribute ("npm", npm);
        return "not-found";
     }
  }
```

Method di atas, akan mengarahkan data berdasarkan select student yang ada di db, dan akan ditampilkan di view form-update, jika tidak ditemukan, akan tampil view not-found, berikut tampilan view all dan form update setelah dipilih





- Tambahkan method dengan tipe request post untuk melakukan update data di studentcontroller

```
@RequestMapping(value = "/student/update/submit", method = RequestMethod.POST)
  public String updateSubmit(@RequestParam(value = "npm", required = false) String npm,
                @RequestParam(value = "name", required = false) String name,
                @RequestParam(value = "gpa", required = false) double gpa)
  {
                StudentModel student = new StudentModel (npm, name, gpa);
      studentDAO.updateStudent(student);

      return "success-update";
  }
```

Berikut hasil setelah melakukan save update submit

Data berhasil diubah/diupdate