A. Pada tutorial 4 ini saya mempelajari mengenani koneksi database dengan Java Spring Boot. Kemudian saya belajar melakukan input delete dan edit data di database. Selain itu saya juga mempelajari pembuatan service update dangan parameter object.
B. Jawaban dari pertanyaan
    1. Menggunakan kode if dengan mengecek apakah variable sudah terisi atau belum.
    2. Karena dengan POST data yang diinputkan lebih aman. Perlu.
    3. Tidek mungkin.
C. Penjelasan kode program delete
    1. Mula-mula saya menambahkan kode untuk membuat sebuah tombol yang nantinya akan digunakan untuk menghapus data tertentu.

```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3     <head>
4         <title>View All Students</title>
5     </head>
6     <body>
7         <h1>All Students</h1>
8 
9         <div th:each="student,iterationStatus: ${students}">
10            <a th:href="'/student/delete/' + ${student.npm}"> Delete Data</a><br/>
11            <a th:href="'/student/update/' + ${student.npm}"> Update Data</a><br/>
12            <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>
13            <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
14            <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
15            <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
16            <hr/>
17        </div>
18    </body>
19 </html>
```

    2. Saya menambahkan method delete di class StudentMapper, pada line atas ada kode SQL untuk menghapus data. Sedangkan line bawah merupakan deklarasi interface untuk method deleteStudent.

```
@Delete("DELETE FROM student WHERE npm = #{npm}")
void deleteStudent (@Param("npm") String npm);
```

    3. Pada class StudentServiceDatabase yang mengimplementasikan interface StudentMapper saya menambahkan kode untuk pendeklarasian method deleteStudent. Line 47 merupakan kode untuk membuat log. Sedangkan line 48 untuk memberikan nilai pada parameter npm kepada method deleteStudent yang berada di class StudentMapper.

```
44     @Override
45     public void deleteStudent (String npm)
46     {
47         log.info("student " + npm + " delete");
48         studentMapper.deleteStudent(npm);
49     }
```

4. Pada class StudentController saya membuat method delete yang akan menghubungkan dan memproses kode untuk menghapus data. Line 93 method delete menampung parameter npm. Line 95 program meminta data berupa object lalu disimpan pada parameter student dari class studentDAO dengan parameter npm. Line 96 program mengecek apakah variabel student berisi nilai object atau null, jika null maka view yang ditampilkan adalah not-found. Namun jika variabel bernilai object maka line 99 akan memproses data ke method deleteStudent yang berada di class studentDAO. Dan pada line 101 program menampilkan view delete.

```
91
92      @RequestMapping("/student/delete/{npm}")
93      public String delete (Model model, @PathVariable(value = "npm") String npm)
94      {
95          StudentModel student = studentDAO.selectStudent(npm);
96          if(student == null) {
97              return "not-found";
98          }
99          studentDAO.deleteStudent (npm);
100
101         return "delete";
102     }
```

D. Penjelasan kode program update

1. Mula-mula saya menambahkan kode untuk membuat sebuah tombol yang nantinya akan digunakan untuk mengedit data tertentu.

```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3     <head>
4         <title>View All Students</title>
5     </head>
6     <body>
7         <h1>All Students</h1>
8
9         <div th:each="student,iterationStatus: ${students}">
10            <a th:href="'/student/delete/' + ${student.npm}"> Delete Data</a><br/>
11            <a th:href="'/student/update/' + ${student.npm}"> Update Data</a><br/>
12            <h3 th:text="'No. ' + ${iterationStatus.count} >No. 1</h3>
13            <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
14            <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
15            <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
16            <hr/>
17        </div>
18    </body>
19 </html>
20
```

2. Saya menambahkan method updateStudent di class StudentMapper, pada line atas ada kode SQL untuk mengedit data. Sedangkan line bawah merupakan deklarasi interface untuk method updateStudent.

```
@Update("UPDATE student SET name = #{name}, gpa = #{gpa} WHERE npm = #{npm}")
void updateStudent (StudentModel student);
```

3. Pada class StudentServiceDatabase yang mengimplementasikan interface StudentMapper saya menambahkan kode untuk pendeklarasian method updateStudent.

```
51      @Override
52      public void updateStudent (StudentModel student) {
53          log.info("student " + student.getNpm() + " update");
54          studentMapper.updateStudent(student);
55      }
```

```
104     @RequestMapping("/student/update/{npm}")
105     public String update (Model model, @PathVariable(value = "npm") String npm)
106     {
107         StudentModel student = studentDAO.selectStudent(npm);
108         if(student == null) {
109             return "not-found";
110         }
111         model.addAttribute ("student", student);
112
113         return "form-update";
114     }
115
116     @RequestMapping(value = "/student/update/submit", method = RequestMethod.POST)
117     public String updateSubmit (StudentModel student)
118     {
119
120         studentDAO.updateStudent (student);
121
122         return "success-update";
123     }
```

E. Penjelasan kode program dengan object sebagai parameter

1. Saya menambahkan method update di class StudentMapper, pada line atas ada kode SQL untuk mengedit data. Sedangkan line bawah merupakan deklarasi interface untuk method updateStudent.

```
@Update("UPDATE student SET name = #{name}, gpa = #{gpa} WHERE npm = #{npm}")
void updateStudent (StudentModel student);
```

2. Pada class StudentServiceDatabase yang mengimplementasikan interface StudentMapper saya menambahkan kode untuk pendeklarasian method updateStudent.

```
51      @Override
52      public void updateStudent (StudentModel student) {
53          log.info("student " + student.getNpm() + " update");
54          studentMapper.updateStudent(student);
55      }
```

[Ar Royya Noor Gunadi Ahmad] [1706106652] [Ekstensi]

```
104     @RequestMapping("/student/update/{npm}")
105     public String update (Model model, @PathVariable(value = "npm") String npm)
106     {
107         StudentModel student = studentDAO.selectStudent(npm);
108         if(student == null) {
109             return "not-found";
110         }
111         model.addAttribute ("student", student);
112 
113         return "form-update";
114     }
115 
116     @RequestMapping(value = "/student/update/submit", method = RequestMethod.POST)
117     public String updateSubmit (StudentModel student)
118     {
119 
120         studentDAO.updateStudent (student);
121 
122         return "success-update";
123     }
```