Disa Ainun Yolanna
1506689654
APAP B

Latihan Menambahkan Delete

1. Pada tutorial sebelumnya Anda sudah menambahkan method delete. Sekarang implementasikan method tersebut menggunakan database.
2. Pada viewall.html tambahkan

```
<a th:href="'/student/delete/'+ ${student.npm}">Delete Data</a>
<hr/>
```

3. Tambahkan method deleteStudent yang ada di class StudentMapper
   a. Tambahkan method delete student yang menerima parameter NPM.
   b. Tambahkan annotation delete di atas dan SQL untuk menghapus

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

4. Lengkapi method deleteStudent yang ada di class StudentServiceDatabase
   a. Tambahkan log untuk method tersebut dengan cara menambahkan
   b. Panggil method delete student yang ada di Student Mapper

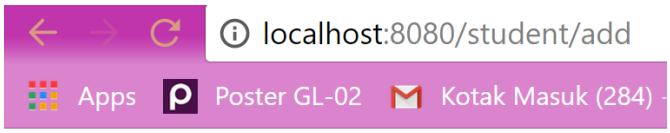
```
@Override
public void deleteStudent (String npm)
{
    log.info("student" + npm + " deleted");
    studentMapper.deleteStudent(npm);
}
```

5. Lengkapi method delete pada class StudentController
   a. Tambahkan validasi agar jika mahasiswa tidak ditemukan tampilkan view not-found
   b. Jika berhasil delete student dan tampilkan view delete

```
@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";
    }
    model.addAttribute( s: "npm",npm);
    return "not-found";
}
```

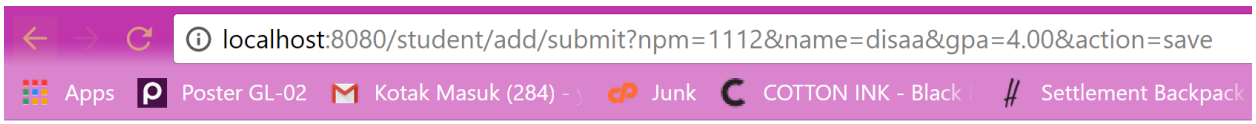
6. Jalankan Spring Boot app dan lakukan beberapa insert

Disa Ainun Yolanna
1506689654
APAP B

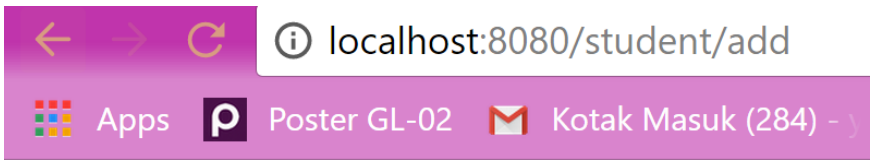


Problem Editor

NPM 1112
Name disaa
GPA 4.00
Save

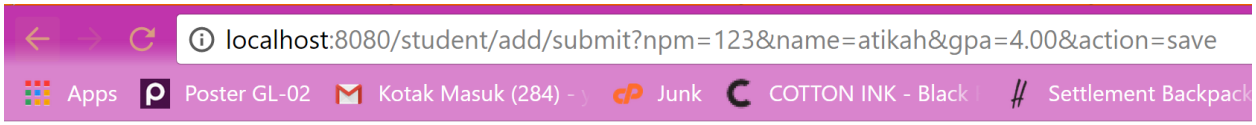


Data berhasil ditambahkan



Problem Editor

NPM 123
Name atikah
GPA 4.00
Save

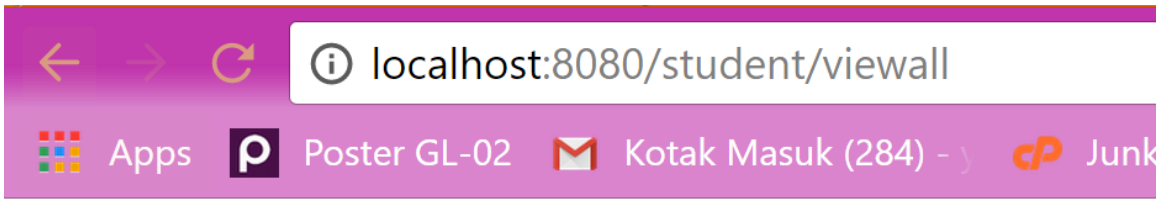


Data berhasil ditambahkan

Disa Ainun Yolanna
1506689654
APAP B

7. Contoh tampilan View All

Disa Ainun Yolanna
1506689654
APAP B



All Students

No. 1

NPM = 1112

Name = disaa

GPA = 4.0

Delete Data

No. 2

NPM = 123

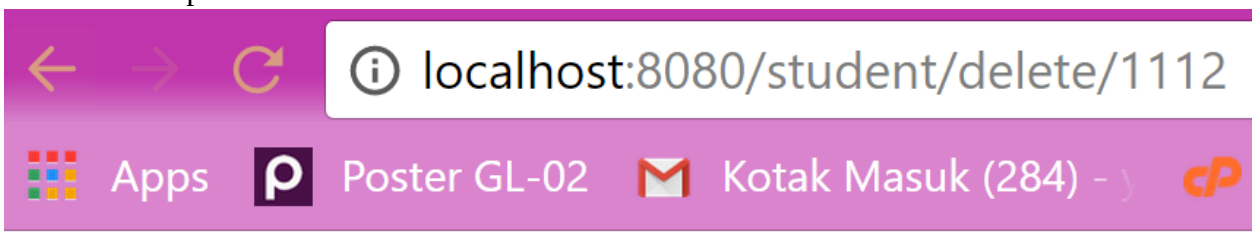
Name = atikah

GPA = 4.0

Delete Data

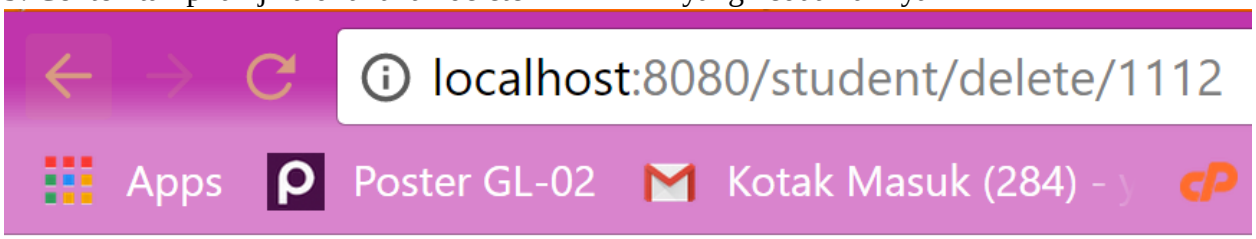
Disa Ainun Yolanna
1506689654
APAP B

8. Contoh tampilan delete NPM 1112



Data berhasil dihapus

9. Contoh tampilan jika dilakukan delete NPM 1112 yang kedua kalinya



Student not found

NPM = 1112

Penjelasan Method Delete: Membuat method deleteStudent pada StudentMapper. Kemudian membuat method deleteStudent pada StudentServiceDarabase, Lalu melanjutkan pembuatan method pada StudentController. Method tsb akan memerima parameter npm dan akan melakukan pengecekan apakah terdapat Student dengan npm tersebut. Jika ditemukan, maka Student dengan NPM tersebut akan terhapus. Jika tidak ditemukan maka penghapusan akan gagal.

Latihan Menambahkan Update

1. Tambahkan method updateStudent pada class StudentMapper
   a. Parameternya adalah StudentModel student

Disa Ainun Yolanna
1506689654
APAP B

b. Annotationnya adalah @Update
c. Lengkapi SQL update-nya

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

2. Tambahkan method updateStudent pada interface StudentService

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

3. Tambahkan implementasi method updateStudent pada class StudentServiceDatabase. Jangan lupa tambahkan logging pada method ini.

```
@Override
public void updateStudent (StudentModel student)
{
    log.info("student name " + student.getName() + " gpa " +student.getGpa() + " updated" );
    studentMapper.updateStudent(student);
```

4. Tambahkan link Update Data pada viewall.html

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

5. Copy view form-add.html menjadi form-update.html.
a. Ubah info-info yang diperlukan seperti title, page-header, tombol menjadi update dll.
b. Ubah action form menjadi /student/update/submit
c. Ubah method menjadi post
d. Untuk input npm, GPA, name.

```
<h1 class="page-header">Update Student</h1>

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

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

6. Copy view success-add.html menjadi success-update.html.
a. Ubah keterangan seperlunya

Disa Ainun Yolanna
1506689654
APAP B

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

7.Tambahkan method update pada class StudentController

a. Request mapping ke /student/update/{npm}

b. Sama halnya seperti delete, lakukan validasi.

c. Jika student dengan npm tidak ada tampilkan view not-found, jika ada tampilkan view form-update

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

        return "form-update";
    }
    model.addAttribute( s: "npm",npm);
    return "not-found";
}
```

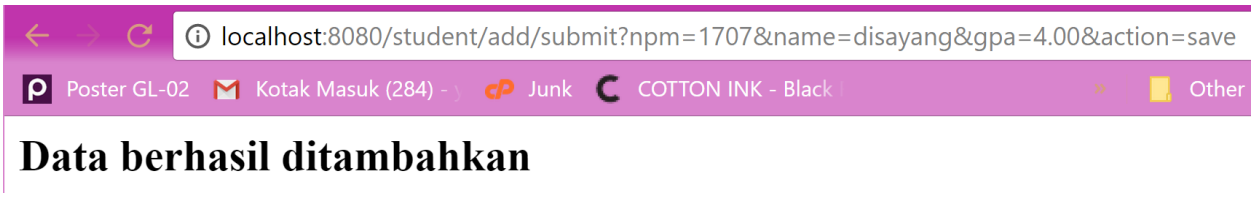
8.Tambahkan method updateSubmit pada class StudentController

```
@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";
}
```

9. Jalankan Spring Boot dan coba test program Anda.

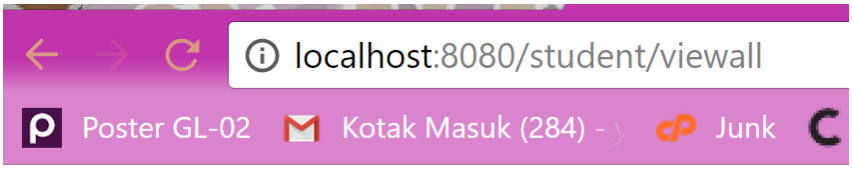
Disa Ainun Yolanna
1506689654
APAP B

Menambah student ke database



Disa Ainun Yolanna
1506689654
APAP B

Data ketika belum diubah



All Students

No. 1

NPM = 1707

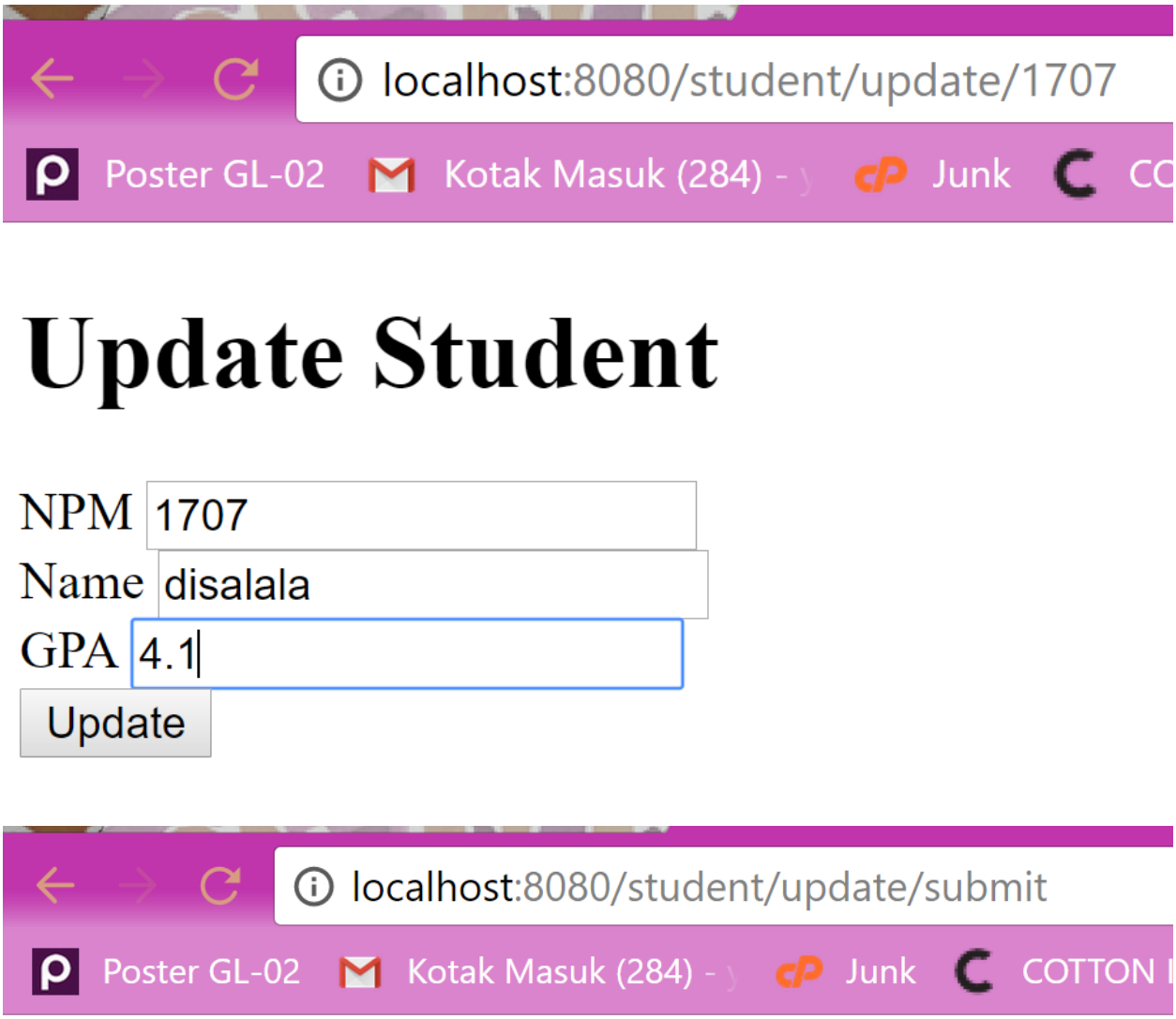
Name = disayang

GPA = 4.0

Delete Data Update Data

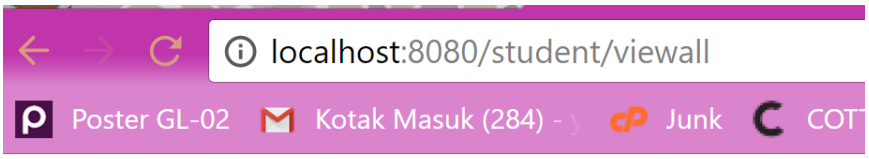
Lakukan perubah data dengan update data

Disa Ainun Yolanna
1506689654
APAP B



Keadaan data setelah berubah

Disa Ainun Yolanna
1506689654
APAP B



All Students

No. 1

NPM = 1707

Name = disalala

GPA = 4.1

Delete Data Update Data

Penjelasan Method Update: Method update akan mengubah nilai yang dimiliki oleh Student. Dengan melakukan pengembalian ke form yang berisi data yang ingin diupdate user. Jika telah melakukan perubahan data, selanjutnya user dapat mengsubmit. Jika dalam pencarian student tidak ditemukan, maka updateStudent gagal.

Latihan Menggunakan Object Sebagai Parameter

Tahapannya kurang lebih sebagai berikut:

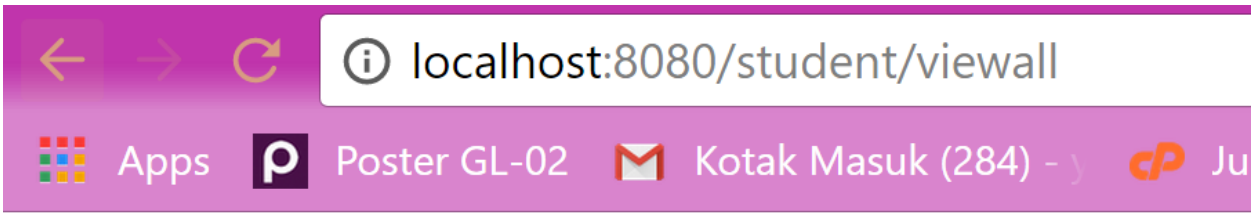
- Menambahkan th:object="${student}" pada tag di view
- Menambahkan th:field="*{[nama_field]}" pada setiap input
- Ubah method updateSubmit pada StudentController yang hanya menerima parameter berupa StudentModel
- Tes lagi aplikasi Anda.

Disa Ainun Yolanna
1506689654
APAP B

```
<body>

<h1 class="page-header">Update Student</h1>

<form th:object="${student}" action="/student/update/submit" method="post">
    <div>
        <label for="npm">NPM</label> <input  type="text" name="npm" readonly="true" th:value="${npm}" th:field="*{[npm]}"/>
    </div>
    <div>
        <label for="name">Name</label> <input type="text" name="name" th:value="${name}" th:field="*{[name]}"/>
    </div>
    <div>
        <label for="gpa">GPA</label> <input type="text" name="gpa" th:value="${gpa}" th:field="*{[gpa]}"/>
    </div>

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

</body>
```

```
@RequestMapping(value = "/student/update/submit", method = RequestMethod.POST)
public String updateSubmit (
        StudentModel student
){
    StudentModel student = new StudentModel (npm, name, gpa);
    studentDAO.updateStudent(student);
    return "success-update";
}
```

Disa Ainun Yolanna
1506689654
APAP B



All Students

No. 1

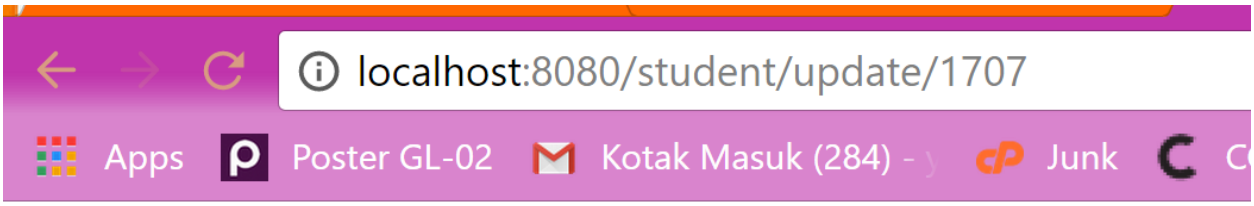
NPM = 1707

Name = disalala

GPA = 4.1

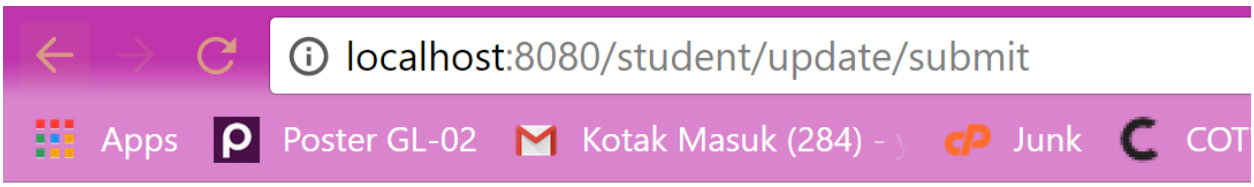
Delete Data Update Data

Disa Ainun Yolanna
1506689654
APAP B



Update Student

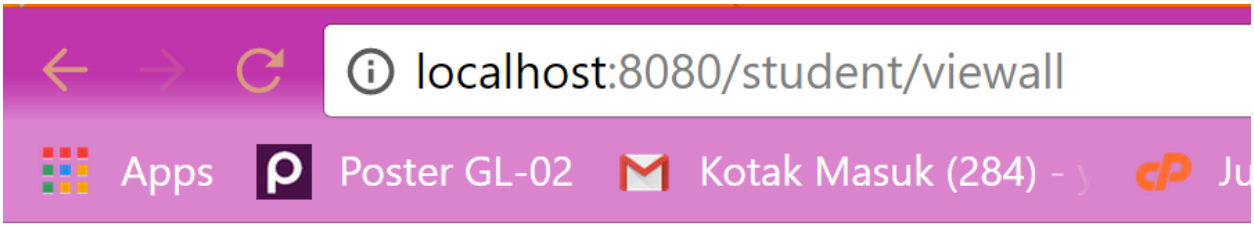
NPM 1707
Name disalinun
GPA 3.9
Update



Data berhasil diupdate

Disa Ainun Yolanna
1506689654
APAP B

:



All Students

No. 1

NPM = 1707

Name = disalinun

GPA = 3.9

Delete Data Update Data

Penjelasan: Pada form-update.html, saya menambahkan th:object dan th:field sehingga nantinya jika form di submit akan dibuat dalam bentuk object akan mempermudah controller. Kemudian, pada updateSubmit, method akan melakukan update Student tanpa membuat object student terlebih dahulu

Pertanyaan Beberapa pertanyaan yang perlu Anda jawab:

1. Jika menggunakan Object sebagai parameter pada form POST, bagaimana caranya melakukan validasi input yang optional dan input yang required seperti jika menggunakan RequestParam? Apakah validasi diperlukan? Asumsikan input pada form Anda tidak menggunakan attribute required sehingga butuh validasi di backend.

Disa Ainun Yolanna
1506689654
APAP B

Jawaban: Hal tersebut memerlukan handling di dalam controller, yang nantinya akan memeriksa apakah semua attribute yang required pada suatu object memiliki nilai atau tidak ketika form telah disubmit. Jika ada atribut yang tidak memiliki nilai maka method di controller akan mengembalikan ke halaman pengisian form

2. Menurut Anda, mengapa form submit biasanya menggunakan POST method dibanding GET method? Apakah perlu penanganan berbeda di header atau body method di controller jika form di post dikirim menggunakan method berbeda?

   Jawaban: Keamanan merupakan salah satu hal yang menjadi alasan mengapa POST method lebih digunakan dibanding GET Method. Diperlukan penyesuaian pada header atau body dari method controller sesuai dengan method yang digunakan

3. Apakah mungkin satu method menerima lebih dari satu jenis request method, misalkan menerima GET sekaligus POST?

   Jawaban: Tidak mungkin.

- Method yang Anda buat pada Latihan Menambahkan Delete
- Method yang Anda buat pada Latihan Menambahkan Update

Method yang Anda buat pada Latihan Menggunakan Object Sebagai Parameter,