

Organizing Domain Logic

v.1.1

PENJELASAN

Halo teman-teman! Setelah belajar mengenai cara menggunakan Git dan melakukan *setup* proyek menggunakan *framework* Spring Boot pada tutorial sebelumnya, sekarang kalian akan memasuki materi berikutnya yang tentunya lebih *asyik*. Pada tutorial sebelumnya, kalian telah mempelajari *basic* dari MVC (Model-View-Controller) dengan membuat proyek Palindrome. Selanjutnya, kalian akan lebih mendalami konsep MVC dengan mempelajari lebih dalam tentang *model* dan *service*.

Model merupakan sebuah objek yang merepresentasikan dan menyimpan informasi terhadap suatu hal. Pada konteks layanan pariwisata, contoh dari model adalah Kebun Safari. Sebuah model pasti memiliki atribut, dalam hal ini objek KebunSafari memiliki atribut seperti nama tempat, alamat, nomor telepon, dan sebagainya.

Service sendiri adalah suatu *layer* yang menjadi mediator antara *controller* dan *database*. Pada *service layer*, tersimpan *business logic* yang digunakan untuk mengolah data yang terdapat dalam *database*. Pengolahan ini meliputi kalkulasi data yang diambil dari *database*, manipulasi *user input* kedalam *database*, dan sebagainya. Contohnya adalah melakukan kalkulasi kuantitas hewan.

PRASYARAT

- DDP2
 - Interface
 - Overriding
- PPW
 - Request Method
 - MVC Framework

LINGKUP PEMBAHASAN

- Model
 - Constructor
 - Setter/getter
- Service
 - Interface

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

- Implementasi Interface
- Controller
 - RequestMethod
 - RequestMapping
 - RequestParam

EKSPEKTASI

- Dapat memahami apa itu *model*, *service*, dan *controller* beserta fungsinya.
- Dapat membuat sebuah *model* dengan konsep MVC dalam project Spring Boot.
- Dapat membuat *service* untuk *create & read* data menggunakan konsep MVC dalam *project* Spring Boot

PENGUMPULAN

- Tenggat Waktu Pengumpulan (Due Date) :
Rabu, 15 September, sebelum 23.55 WIB untuk semua kelas. Setiap keterlambatan selama 24 jam akan mengakibatkan -20% dari nilai sebenarnya.
- Pengumpulan tutorial dalam bentuk **Pull Request (PR)** dari **branch feat/tutorial-2** ke **master**. Waktu keterlambatan dilihat dari **aktivitas perubahan** di Pull Request.
- Jawablah **semua** pertanyaan dan kerjakan **semua** latihan yang ada pada tutorial.
- Jawablah pertanyaan-pertanyaan yang ada pada file **README.md** yang **sama** seperti tutorial sebelumnya.
- Tutorial 2 di-*push* ke dalam *repository* yang sama dengan tutorial sebelumnya sehingga pada *repository* akan terdapat dua folder yaitu *isPalindrome* dan *kebunSafari*.
- Gunakan teknik *branching* dan PR yang sama seperti pada tutorial sebelumnya.
- Tutorial 2 **ada demo**, silahkan isi slot demo di SCELE.

PENILAIAN

$\text{Nilai akhir} = \text{Nilai Tutorial} + \text{Latihan} + \text{Pertanyaan} + \text{Demo}$

Plagiarisme akan dikenakan sanksi nilai 0! ☹

- Indikator Penilaian Tutorial
 - Kesesuaian Model, Controller, Repository, dan View Templates.
 - Keberhasilan Proyek KebunSafari.
 - Kelengkapan dan Ketepatan Jawaban pada README.md
- Demo

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

PRA-TUTORIAL

1. Pastikan kamu sudah melakukan **Squash & Merge** dari branch **tutorial** sebelumnya ke **master**
2. Sebelum mulai mengerjakan tutorial ini, buat **issue "Pengerjaan Tutorial 2"** pada Repo GitHub anda
3. Pastikan pada lokal kamu sudah pindah ke **branch master**, kemudian **pull master**
4. Buat **branch baru** dari **Master**, dengan nama **'feat/tutorial-2-kebun-safari'**

PASCA-TUTORIAL

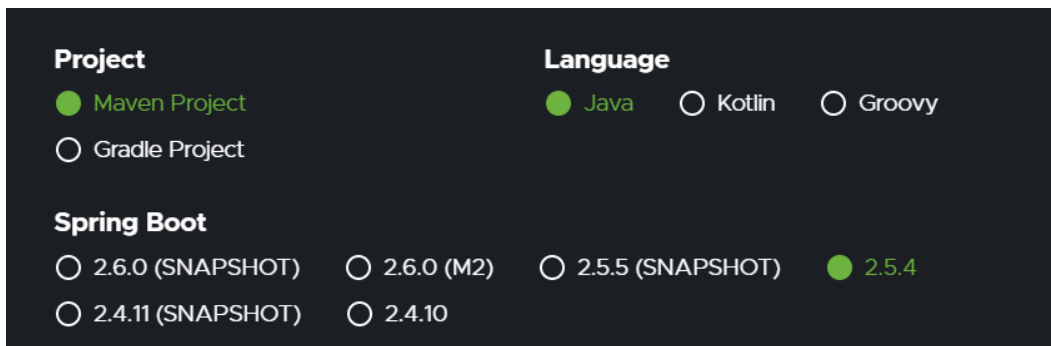
1. Pull Request dengan meminta *review* dari asisten melalui GitHub sebelum **Deadline**
2. Isi slot demo sesuai dengan pembagian asisten minggu ini, slot demo akan dibuka pada **Rabu 15 September Pukul 20.00 WIB**.
3. Asisten akan melakukan *approve* Pull Request setelah demo dilaksanakan

TUTORIAL

Pada kondisi pandemi seperti ini, Papa APAP memanggilmu untuk berbincang-bincang mengenai suatu hal yang sangat penting. Kamu pun memenuhi panggilan tersebut. Papa APAP bercerita bahwa ia ingin sekali membuat sebuah *travel-tech service* bernama Kebun Safari. Sebenarnya, Papa APAP bisa saja membuatnya sendiri, namun Papa APAP ingin anaknya dapat mengalami sendiri bagaimana serunya mengembangkan sebuah *enterprise application*. Sebagai anak yang baik, tentunya kamu akan menuruti dan menyetujui permintaan Papa APAP. Sebagai awal mula dari project Kebun Safari, kamu akan mencoba menerapkan ilmu yang kamu telah pelajari di kelas dengan membuat sebuah *base project* MVC yang sangat sederhana dan kamu yakin bahwa kamu bisa mengerjakannya dengan baik.

Inisiasi Proyek

1. Buatlah sebuah Spring project baru melalui <https://start.spring.io/>
2. Isikan form sesuai gambar berikut



The screenshot shows the Spring Initializr web form. It has a dark background with white text. The form is divided into three sections: Project, Language, and Spring Boot. In the Project section, 'Maven Project' is selected with a green radio button. In the Language section, 'Java' is selected with a green radio button. In the Spring Boot section, '2.5.4' is selected with a green radio button.

Project		Language		
<input checked="" type="radio"/> Maven Project		<input checked="" type="radio"/> Java	<input type="radio"/> Kotlin	<input type="radio"/> Groovy
<input type="radio"/> Gradle Project				

Spring Boot			
<input type="radio"/> 2.6.0 (SNAPSHOT)	<input type="radio"/> 2.6.0 (M2)	<input type="radio"/> 2.5.5 (SNAPSHOT)	<input checked="" type="radio"/> 2.5.4
<input type="radio"/> 2.4.11 (SNAPSHOT)	<input type="radio"/> 2.4.10		

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

Project Metadata

Group	apap.tutorial
Artifact	kebunsafari
Name	kebunsafari
Description	Project KebunSafari APAP 2021
Package name	apap.tutorial.kebunsafari
Packaging	<input checked="" type="radio"/> Jar <input type="radio"/> War
Java	<input type="radio"/> 16 <input checked="" type="radio"/> 11 <input type="radio"/> 8

3. Tambahkan *dependencies* berikut

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Boot DevTools

DEVELOPER TOOLS

Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Spring Web

WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Thymeleaf

TEMPLATE ENGINES

A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes.

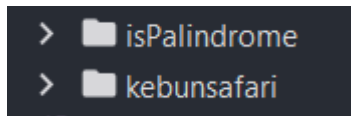
4. Klik pada “Generate the project” untuk membuat Spring project.

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

5. Ekstrak *project* yang telah kamu *generate* ke *root directory* seperti pada tutorial sebelumnya. Sehingga *root directory* didalam repo kamu memiliki struktur folder seperti berikut



Model

Salah satu komponen dalam MVC adalah model. Kamu akan membuat model berupa class KebunSafari yang berisi variabel idKebunSafari (String), namaKebunSafari (String), alamat (String), noTelepon (String).

1. Klik kanan pada **Project** > **New** > **Package**. Buatlah *package* **apap.tutorial.kebunsafari.model**
2. Pada *package* tersebut, buatlah *class* **KebunSafariModel** dengan spesifikasi seperti berikut

```
package apap.tutorial.kebunsafari.model;  
  
public class KebunSafariModel {  
    private String idKebunSafari;  
    private String namaKebunSafari;  
    private String alamat;  
    private String noTelepon;  
}
```

3. Tambahkan constructor, setter, dan getter.

Tips: *Constructor, setter, dan getter* dapat dibuat secara otomatis melalui menu:

- **STS:** Source > Generate Getters and Setters... dan Source > Generate Constructor using Fields...
- **IntelliJ:**
 - Code > Generate > Constructor lalu pilih semua field dan tekan OK
 - Code > Generate > Getter and Setter lalu pilih semua field dan tekan OK

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

Service

Setelah membuat *model*, kamu juga akan membuat sebuah *service* untuk mendefinisikan *method-method* apa saja yang dapat dijalankan untuk memanipulasi *class* KebunSafari. Di dalam *service* sendiri, terdapat dua jenis *service* yang dapat dibuat, yaitu *service* yang berbentuk *interface* dan *service* yang mengimplementasikan *interface* itu sendiri.

1. Klik kanan pada Project > New > Package. Buatlah *package* **apap.tutorial.kebunsafari.service**
2. Pada *package* tersebut, buatlah *interface* **KebunSafariService** dengan spesifikasi seperti berikut.

```
package apap.tutorial.kebunsafari.service;

import apap.tutorial.kebunsafari.model.KebunSafariModel;

import java.util.List;

public interface KebunSafariService {
    // Method untuk menambahkan KebunSafari baru
    void addKebunSafari(KebunSafariModel kebunSafari);

    // Method untuk mendapatkan seluruh daftar KebunSafari
    List<KebunSafariModel> getKebunSafariList();

    // Method untuk mendapatkan data sebuah KebunSafari berdasarkan ID yang dimiliki
    KebunSafariModel getKebunSafariByIdKebunSafari(String idKebunSafari);
}
```

3. Setelah membuat *interface*, buatlah sebuah class bernama KebunSafariInMemoryService yang akan mengimplementasikan *interface* KebunSafariService itu sendiri pada package yang sama (**apap.tutorial.kebunsafari.service**)

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

```
package apap.tutorial.kebunsafari.service;

import apap.tutorial.kebunsafari.model.KebunSafariModel;
import org.springframework.stereotype.Service;
import java.util.List;
import java.util.ArrayList;

@Service
public class KebunSafariInMemoryService implements KebunSafariService {
    private List<KebunSafariModel> listKebunSafari;

    // Constructor
    public KebunSafariInMemoryService(){
        listKebunSafari = new ArrayList<>();
    }

    @Override
    public void addKebunSafari(KebunSafariModel kebunSafari) {
        listKebunSafari.add(kebunSafari);
    }

    @Override
    public List<KebunSafariModel> getKebunSafariList() {
        return listKebunSafari;
    }
}
```

Pada *screenshot* di atas, hanya 2 *method* yang baru kamu terapkan. **Kerjakan** *method* `getKebunSafariByIdKebunSafari` sebelum melanjutkan ke step berikutnya.

Tips:

- Manfaatkan `listKebunSafari` sebagai searching space kamu, gunakan `idKebunSafari` sebagai kriteria untuk mencari.
- `KebunSafariInMemoryService` menyimpan/mengambil data-data Kebun dari sebuah *static List* dan tidak melakukan koneksi ke *database*. Pada tutorial berikutnya kamu akan mempelajari bagaimana cara melakukan *service* CRUD pada *database*.
- Pastikan penggunaan anotasi '@' tidak ada yang terlewatkan seperti *screenshot* diatas

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

Controller

Controller berfungsi sebagai *request handler*. *Controller* akan melakukan *mapping* dari sebuah *request* yang diinput oleh *user* untuk memanggil *service* yang kita inginkan.

1. Klik kanan pada Project > New > Package. Buatlah package **apap.tutorial.kebunsafari.controller**
2. Pada package tersebut, buatlah class **KebunSafariController** dengan spesifikasi seperti berikut:

```
@Controller
public class KebunSafariController {
    @Autowired
    private KebunSafariService kebunSafariService;

    @RequestMapping("/kebun-safari/add")
    public String addKebunSafari(
        @RequestParam (value="id", required = true) String idKebunSafari,
        @RequestParam (value="nama", required = true) String namaKebunSafari,
        @RequestParam (value="alamat", required = true) String alamat,
        @RequestParam (value="noTelepon", required = true) String noTelepon,
        Model model
    ){
        // Membuat Objek Kebun Safari baru
        KebunSafariModel kebunSafari = new KebunSafariModel(idKebunSafari, namaKebunSafari, alamat, noTelepon);

        // Memanggil service addKebunSafari
        kebunSafariService.addKebunSafari(kebunSafari);

        // Menambahkan variabel kebunSafari untuk dirender di Thymeleaf
        model.addAttribute(s: "kebunSafari", kebunSafari);

        // Mereturn template html yang dipakai
        return "add-kebun-safari";
    }
}
```

Pertanyaan 1: Cobalah untuk menambahkan sebuah Kebun dengan mengakses link berikut setelah menjalankan program:

<http://localhost:8080/kebun-safari/add?id=1&nama=Papa%20APAP&alamat=Maung%20Fasilkom&noTelepon=081xxx> Apa yang terjadi? Jelaskan mengapa hal tersebut dapat terjadi

Pertanyaan 2: Menurut kamu anotasi `@Autowired` pada class Controller tersebut merupakan implementasi dari konsep apa? Dan jelaskan secara singkat cara kerja `@Autowired` tersebut dalam konteks service dan controller yang telah kamu buat

3. Tambahkan *handler* lainnya untuk meng-handle sebuah *request* yang akan memanggil method `getKebunSafariList` dan `getKebunSafariByIdKebunSafari`. Ingat, pastikan kamu telah

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

membuat implementasi *service* dari *getKebunSafariByIdKebunSafari* pada tahap sebelumnya.

getKebunSafariList

```
@RequestMapping("/")
public String listKebunSafari(Model model){
    // Mendapatkan list seluruh objek Kebun Safari
    List<KebunSafariModel> listKebunSafari = kebunSafariService.getKebunSafariList();

    // Menambahkan list untuk dirender di Thymeleaf
    model.addAttribute(s: "listKebunSafari", listKebunSafari);

    // Mereturn template html yang dipakai
    return "get-all-kebun-safari";
}
```

getKebunSafariByIdKebunSafari

```
@RequestMapping("/kebun-safari")
public String getKebunSafariById(@RequestParam(value="id", required = true) String idKebunSafari, Model model){
    // Mendapatkan Objek Kebun Safari sesuai dengan ID
    KebunSafariModel kebunSafari = kebunSafariService.getKebunSafariByIdKebunSafari(idKebunSafari);

    // Menambahkan objek untuk dirender di Thymeleaf
    model.addAttribute(s: "kebunSafari", kebunSafari);

    // Mereturn template html yang dipakai
    return "detail-kebun-safari";
}
```

View Template (HTML & Thymeleaf)

Pada step 2 bagian *Controller* kamu pasti akan menemukan sebuah *error* saat mengakses link yang diberikan. Hal tersebut terjadi karena *view* yang telah kamu cantumkan pada *Controller* ("add-kebun-safari") belum dibuat. Pada tutorial ini kamu akan belajar membuat *view template* menggunakan *Thymeleaf*. *Thymeleaf* berfungsi untuk menampilkan *model* yang dirender oleh *Front Controller* kepada *user*.

1. Pada directory **resources/templates** tambahkan **add-kebun-safari.html** dengan spesifikasi sebagai berikut.

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

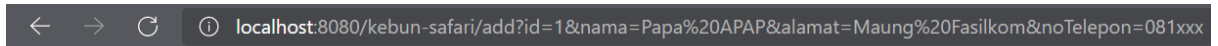
Deadline: 15 September 2021, 23.55 waktu Scele

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://thymeleaf.org">
<head>
  <title>Tambah Kebun Safari</title>
</head>
<body>
  <h2 th:text="'Kebun Safari dengan ID ' + ${kebudSafari.idKebunSafari} + ' berhasil ditambahkan'"></h2>
  <a href="/">Kembali</a>
</body>
</html>
```

View template dengan nama “**add-kebun-safari.html**” telah berhasil kamu buat. Jalankan program dan akses kembali link berikut:

<http://localhost:8080/kebun-safari/add?id=1&nama=Papa%20APAP&alamat=Maung%20Fasilkom&noTelepon=081xxx>

Jika kamu telah berhasil menampilkan halaman seperti dibawah ini, **HORE!** Kamu telah berhasil mengimplementasikan salah satu method (addKebunSafari) dengan baik.



Kebun Safari dengan ID 1 berhasil ditambahkan

[Kembali](#)

Pertanyaan 3: Cobalah untuk menambahkan sebuah Kebun dengan mengakses link berikut:

<http://localhost:8080/kebun-safari/add?id=1&nama=Papa%20APAP&alamat=Maung%20Fasilkom> Apa yang terjadi? Jelaskan mengapa hal tersebut dapat terjadi.

2. Pada *directory* yang sama tambahkan *view template* lainnya yang belum ditambahkan (detail-kebun-safari.html dan get-all-kebun-safari.html) dengan spesifikasi sebagai berikut

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

detail-kebun-safari.html

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://thymeleaf.org">
<head>
  <title>Detail Kebun Safari</title>
</head>
<body>
  <h2>Detail Kebun Safari</h2>
  <p th:text="' ID Kebun Safari : ' + ${kebunSafari.idKebunSafari} "></p>
  <p th:text="' Nama Kebun Safari : ' + ${kebunSafari.namaKebunSafari} "></p>
  <p th:text="' Alamat : ' + ${kebunSafari.alamat} "></p>
  <p th:text="' Nomor Telepon : ' + ${kebunSafari.noTelepon} "></p>
  <a href="/">Kembali</a>
</body>
</html>
```

get-all-kebun-safari.html

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://thymeleaf.org">
<head>
  <title>Daftar Seluruh Kebun Safari</title>
</head>
<body>
  <h2>Daftar seluruh Kebun Safari</h2>
  <div th:each="kebunSafari : ${listKebunSafari}">
    <h3 th:text="' ID Kebun Safari : ' + ${kebunSafari.idKebunSafari} "></h3>
    <p th:text="' Nama Kebun Safari : ' + ${kebunSafari.namaKebunSafari} "></p>
    <p th:text="' Alamat : ' + ${kebunSafari.alamat} "></p>
    <p th:text="' Nomor Telepon : ' + ${kebunSafari.noTelepon} "></p>
  </div>
</body>
</html>
```

Pertanyaan 4: Jika Papa APAP ingin melihat Kebun Safari dengan nama Papa APAP, link apa yang harus diakses?

Pertanyaan 5: Tambahkan 1 contoh Kebun Safari lainnya sesukamu. Lalu cobalah untuk mengakses <http://localhost:8080/> , apa yang akan ditampilkan? Sertakan juga bukti screenshotmu.

Tutorial 2 Arsitektur & Pemrograman Aplikasi Perusahaan

Semester Ganjil Tahun Ajaran 2021/2022

Deadline: 15 September 2021, 23.55 waktu Scele

Tips: Silahkan mencoba-coba untuk mengakses semua mapping yang telah kamu buat. Pastikan semuanya sudah berjalan dengan lancar dan menampilkan sesuai yang diminta.

LATIHAN

1. Pada *KebunSafariController* tambahkan sebuah *method view* Kebun dengan menggunakan **Path Variable**. Misalnya, kamu ingin memasukkan data sebuah Kebun yang memiliki **idKebunSafari** 1, untuk melihat data yang baru dimasukkan tersebut, *user* dapat mengakses halaman <http://localhost:8080/kebun-safari/view/1>.
2. Tambahkan fitur untuk melakukan **update noTelepon** Kebun berdasarkan **idKebunSafari**. Misalnya, setelah melakukan *add* Kebun pada tahap 1 bab *View Template*, cobalah untuk mengubah **noTelepon** objek **KebunSafari** tersebut menjadi “021752xxxx” dengan mengakses halaman <http://localhost:8080/kebun-safari/update/1?noTelepon=021752xxxx>
Tampilkan juga sebuah **halaman** yang memberikan informasi bahwa data tersebut telah **berhasil diubah**.
3. Tambahkan fitur untuk melakukan **delete** Kebun berdasarkan **idKebunSafari**. Misalnya, setelah melakukan *add* Kebun pada tahap 1 bab *View Template* dan melakukan *update* seperti pada latihan nomor 2, cobalah untuk melakukan *delete* data tersebut dengan mengakses halaman <http://localhost:8080/kebun-safari/delete/1>. **Tampilkan** sebuah **halaman** yang memberikan informasi bahwa data tersebut telah **berhasil dihapus**.

Notes: Jika **idKebunSafari** tidak diberikan atau tidak ditemukan kembalikan halaman error yang berisi informasi bahwa **idKebunSafari** kosong atau tidak ditemukan dan proses *view/update/delete* dibatalkan