LAPORAN ANTARA (PROGRESS REPORT) APLIKASI MANAJEMEN DATA PASIEN (PATIENTAPP)

Periode Laporan: Minggu ke-2 hingga Minggu ke-3 Status proyek: On-Schedule (Sesuai Jadwal) Presentase Penyelesaian: 75%

Disusun Oleh :	
M. Afwan Sudiro	(50422973)
Nofendra Tahta Dirgantara	(51422249)
Raihan Musyaffa Hanif	(51422357)

JAKARTA

2026

DAFTAR ISI

BAB I PENDAHULUAN **3**
1.1 Latar Belakang 3
1.2 Tujuan Pengembangan 3
1.3 Batasan Masalah 3
BAB II LANDASAN TEORI **4**
2.1 Spring Boot 4
2.2 Model-View-Controller (MVC) 4
2.3 Spring Data JPA 4
BAB III ANALISIS DAN PERANCANGAN SISTEM **5**
3.1 Analisis Kebutuhan Sistem 5
Kebutuhan Non-Fungsional 5
3.2 Arsitektur Sistem 5
3.3 Diagram Alur MVC (Deskriptif) 5
BAB IV **6**
IMPLEMENTASI SISTEM (HASIL SEMENTARA) **6**
4.1 Struktur Folder Aplikasi 6
4.2 Implementasi Model 7
4.3 Implementasi Repository 7
4.4 Implementasi Controller 8
4.5 Hasil Sementara 8
BAB V KESIMPULAN SEMENTARA **10**
5.1 Kesimpulan 10
5.2 Rencana Pengembangan Selanjutnya 10

BAB I PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi di bidang kesehatan menuntut adanya sistem informasi yang mampu mengelola data pasien secara efektif, terstruktur, dan aman. Pengelolaan data pasien secara manual berpotensi menimbulkan kesalahan pencatatan, kehilangan data, serta keterlambatan dalam penyajian informasi.

Oleh karena itu, dikembangkan sebuah aplikasi berbasis web bernama PatientApp yang bertujuan untuk membantu proses pengelolaan data pasien secara terkomputerisasi dengan memanfaatkan framework Spring Boot dan konsep Model-View-Controller (MVC).

1.2 Tujuan Pengembangan

Tujuan dari pengembangan aplikasi PatientApp adalah:

1. Membangun aplikasi manajemen data pasien berbasis web.
2. Menerapkan arsitektur MVC (Model-View-Controller).
3. Mengimplementasikan koneksi database menggunakan Spring Data JPA.
4. Menyediakan fitur CRUD (Create, Read, Update, Delete) data pasien.
5. Melatih penerapan konsep pemrograman berbasis framework Java.

1.3 Batasan Masalah

Dalam laporan antara ini, pembahasan dibatasi pada:

- Pengelolaan data pasien dasar (nama, NIK, dll).
- Implementasi backend menggunakan Spring Boot.
- Penggunaan database relasional melalui JPA Repository.
- Tidak membahas aspek keamanan tingkat lanjut (authentication & authorization).

BAB II LANDASAN TEORI

2.1 Spring Boot

Spring Boot merupakan framework Java yang digunakan untuk mempermudah pengembangan aplikasi berbasis Spring dengan konfigurasi minimal. Spring Boot menyediakan embedded server, dependency management, dan integrasi yang baik dengan Spring Data JPA.

2.2 Model-View-Controller (MVC)

MVC adalah pola arsitektur yang memisahkan aplikasi menjadi tiga komponen utama:

- **Model**: Mengelola data dan logika bisnis.
- **View**: Menampilkan data ke pengguna.
- **Controller**: Menghubungkan Model dan View serta menangani request.

2.3 Spring Data JPA

Spring Data JPA digunakan untuk mempermudah proses akses database dengan pendekatan repository, sehingga developer tidak perlu menulis query SQL secara manual.

BAB III ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Kebutuhan Sistem

Kebutuhan Fungsional

- Menambah data pasien
- Menampilkan daftar pasien
- Mengubah data pasien
- Menghapus data pasien

Kebutuhan Non-Fungsional

- Aplikasi berbasis web
- Menggunakan Java dan Spring Boot
- Database terintegrasi
- Mudah digunakan dan dikembangkan

3.2 Arsitektur Sistem

Aplikasi PatientApp menggunakan arsitektur MVC, dengan struktur sebagai berikut:

Komponen	Deskripsi
Model	Patient.java
Repository	PatientRepository.java
Controller	PatientController.java
Main App	PatientAppApplication.java

3.3 Diagram Alur MVC (Deskriptif)

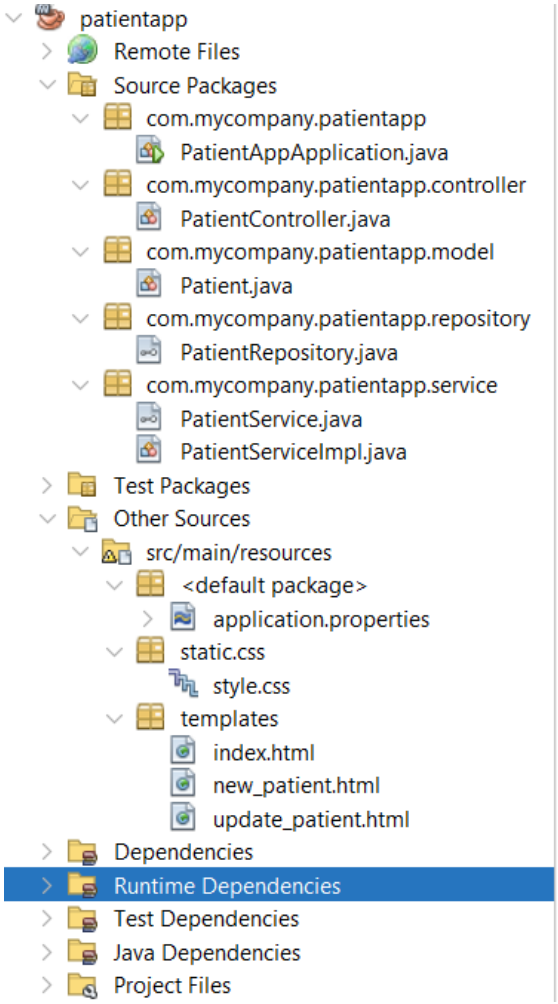
1. User mengirim request melalui browser.
2. Request diterima oleh PatientController.
3. Controller memanggil PatientRepository untuk akses data.
4. Data diproses melalui Model Patient.
5. Hasil dikirim kembali ke View untuk ditampilkan.

BAB IV

IMPLEMENTASI SISTEM (HASIL SEMENTARA)

4.1 Struktur Folder Aplikasi

Struktur utama program:



4.2 Implementasi Model

Kelas Patient berfungsi sebagai representasi tabel pasien di database dan berisi atribut-atribut data pasien.

Patient.java

```java
package com.mycompany.patientapp.model;

import jakarta.persistence.*;
import lombok.*;
import org.springframework.format.annotation.DateTimeFormat;
import java.time.LocalDate;

@NoArgsConstructor
@AllArgsConstructor
@Data
@Entity
@Table(name = "patients")
public class Patient {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(nullable = false, length = 20)
    private String nik;

    @Column(nullable = false)
    private String nama;

    @Column(name = "jenis_kelamin")
    private String jenisKelamin; // Laki-laki / Perempuan

    @Column(name = "tanggal_lahir")
    @DateTimeFormat(pattern = "yyyy-MM-dd") // Format input HTML5
    private LocalDate tanggalLahir;

    @Column(columnDefinition = "TEXT")
    private String alamat;

    @Column(name = "nomor_telepon")
    private String nomorTelepon;

    @Column(columnDefinition = "TEXT")
    private String keluhan;
}
```

4.3 Implementasi Repository

PatientRepository menggunakan JpaRepository untuk menyediakan operasi CRUD secara otomatis tanpa query manual.

PatientRepository.java

```java
package com.mycompany.patientapp.repository;

import com.mycompany.patientapp.model.Patient;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface PatientRepository extends JpaRepository<Patient, Long> {
}
```

4.4 Implementasi Controller

PatientController bertugas:

- Menerima request HTTP
- Mengatur alur tambah, simpan, dan tampil data pasien
- Menghubungkan Model dan View

PatientController.java

Source | History

```java
import com.mycompany.patientapp.service.PatientService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@Controller
public class PatientController {

    @Autowired
    private PatientService patientService;

    // Halaman Utama
    @GetMapping("/")
    public String viewHomePage(Model model) {
        return findPaginated(1, "nama", "asc", model);
    }

    @GetMapping("/showNewPatientForm")
    public String showNewPatientForm(Model model) {
        // Membuat objek model atribut untuk mengikat data form
        Patient patient = new Patient();
        model.addAttribute("patient", patient);
        return "new_patient";
    }

    @PostMapping("/savePatient")
    public String savePatient(@ModelAttribute("patient") Patient patient) {
        // Simpan pasien ke database
        patientService.savePatient(patient);
        return "redirect:/";
    }

    @GetMapping("/showFormForUpdate/{id}")
    public String showFormForUpdate(@PathVariable(value = "id") long id, Mode
        // Ambil pasien dari service
        Patient patient = patientService.getPatientById(id);

        // Set pasien sebagai model atribut untuk pre-populate form
        model.addAttribute("patient", patient);
        return "update_patient";
    }

    @GetMapping("/deletePatient/{id}")
    public String deletePatient(@PathVariable(value = "id") long id) {
        // Panggil method delete pada service
```

4.5 Hasil Sementara

- Aplikasi berhasil dijalankan menggunakan Spring Boot.
- Proses tambah dan simpan data pasien sudah berjalan.
- Koneksi database berhasil menggunakan JPA.
- Struktur MVC telah diterapkan dengan baik.

DATA PASIEN BARU

NIK

Masukkan NIK

Nama Lengkap

Nama Pasien

Jenis Kelamin

Laki-laki

Tanggal Lahir

dd/mm/yyyy

Alamat

Alamat Lengkap

Nomor Telepon

Contoh: 08123456789

Keluhan Utama

Ceritakan keluhan yang dirasakan...

Simpan

← Kembali ke Daftar

DAFTAR PASIEN

Tambah Pasien

NIK	NAMA PASIEN	JENIS KELAMIN	TGL LAHIR	TELEPON	KELUHAN	AKSI
178345667889990235	Albert Gabriel	Laki-laki	2003-07-06	081567889999	Perut terasa sakit nyeri dan tidak bisa buang angin	Edit Hapus

BAB V KESIMPULAN SEMENTARA

5.1 Kesimpulan

Berdasarkan hasil implementasi sementara, dapat disimpulkan bahwa:

1. Aplikasi PatientApp berhasil menerapkan arsitektur MVC.
2. Spring Boot dan Spring Data JPA mempermudah pengembangan aplikasi.
3. Fitur CRUD data pasien telah berjalan dengan baik.
4. Aplikasi layak untuk dikembangkan lebih lanjut.

5.2 Rencana Pengembangan Selanjutnya

- Penambahan fitur validasi input
- Implementasi tampilan (Thymeleaf) yang lebih interaktif
- Penambahan fitur pencarian data pasien
- Implementasi keamanan sistem (login & role)