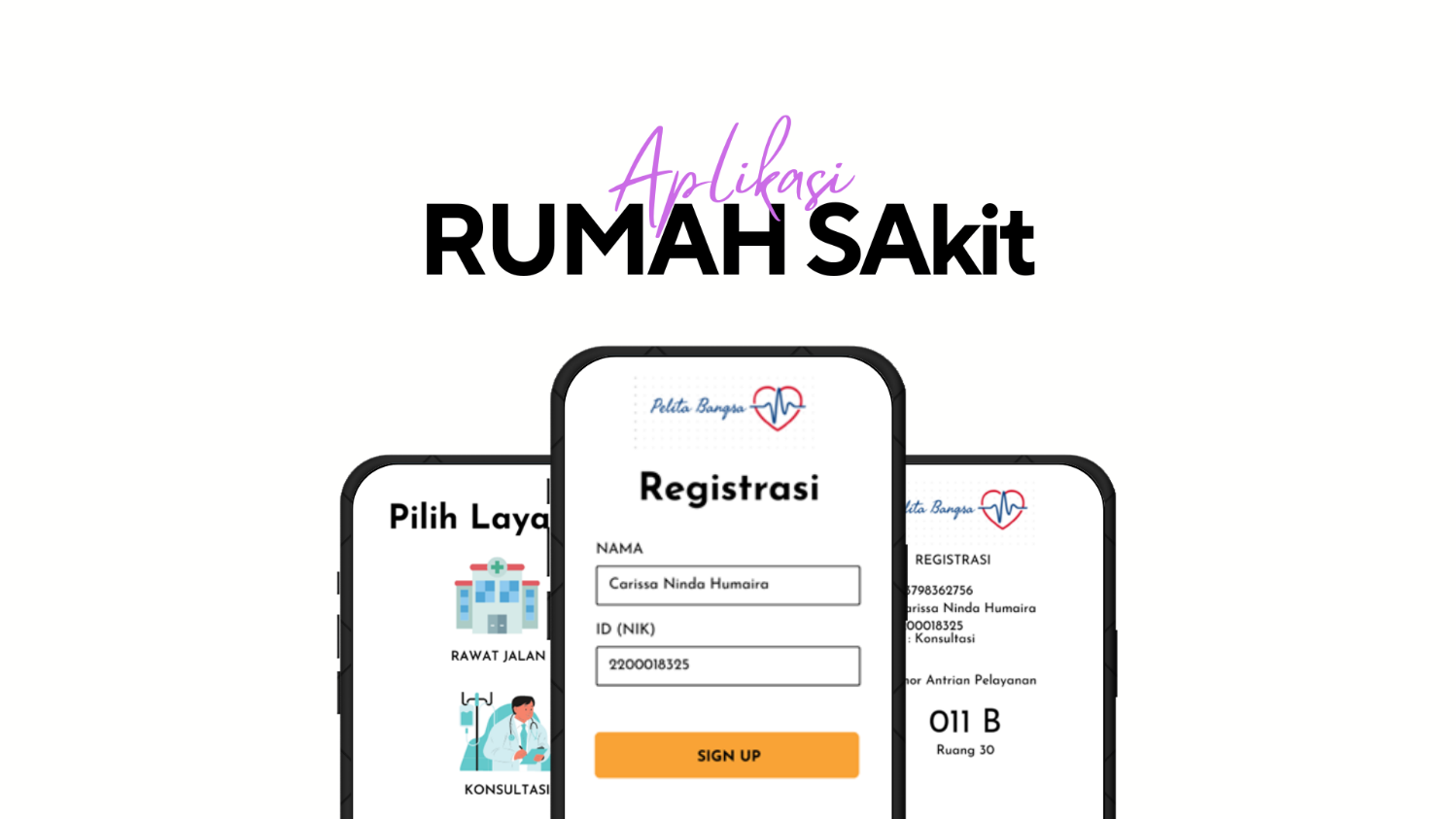
**TUGAS AKHIR SEMESTER 3**

**PEMROGRAMAN BERBASIS OBJEK**

**“Aplikasi Sederhana Rumah Sakit”**



DISUSUN OLEH :

NAMA : CARISSA NINDA HUMAIRA

NIM : 2200018325

KELAS : G

Github : <https://github.com/users/carissaninda11/projects/2>

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS TEKNOLOGI INDUSTRI**

**UNIVERSITAS AHMAD DAHLAN**

**JANUARI 2024**

1. **Deskripsi pembuatan aplikasi Registrasi Rumah Sakit**

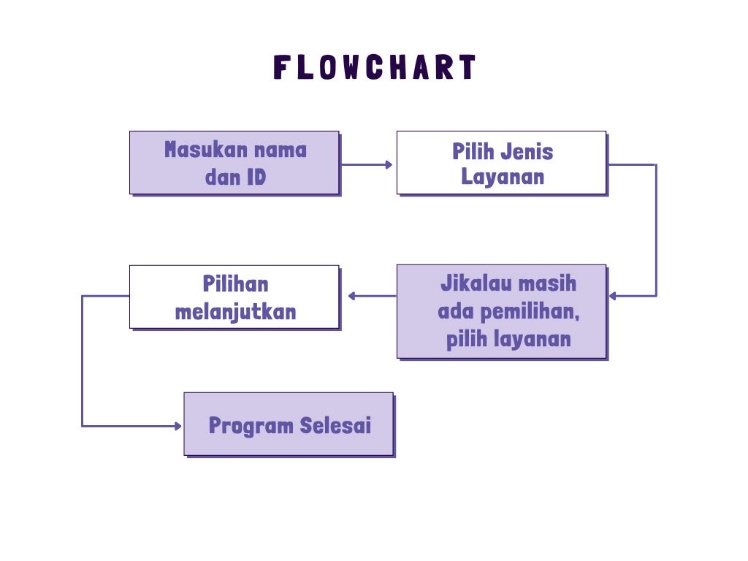
Sangat sering terjadi Rumah sakit mengalami kesulitan untuk melayani Registrasi atau pendaftaran pasien, kesulitan tersebut adalah pasien menunggu terlalu lama untuk mendapatkan pelayanan padahal seharusnya pelayanan yang berhubungan dengan kesehatan harus dilakukan dengan cepat. Selain itu biasanya pendaftaran dilakukan ketika pasien sudah berada di rumah sakit dan harus mengantri lagi untuk melakukan pendaftaran secara manual di kertas.

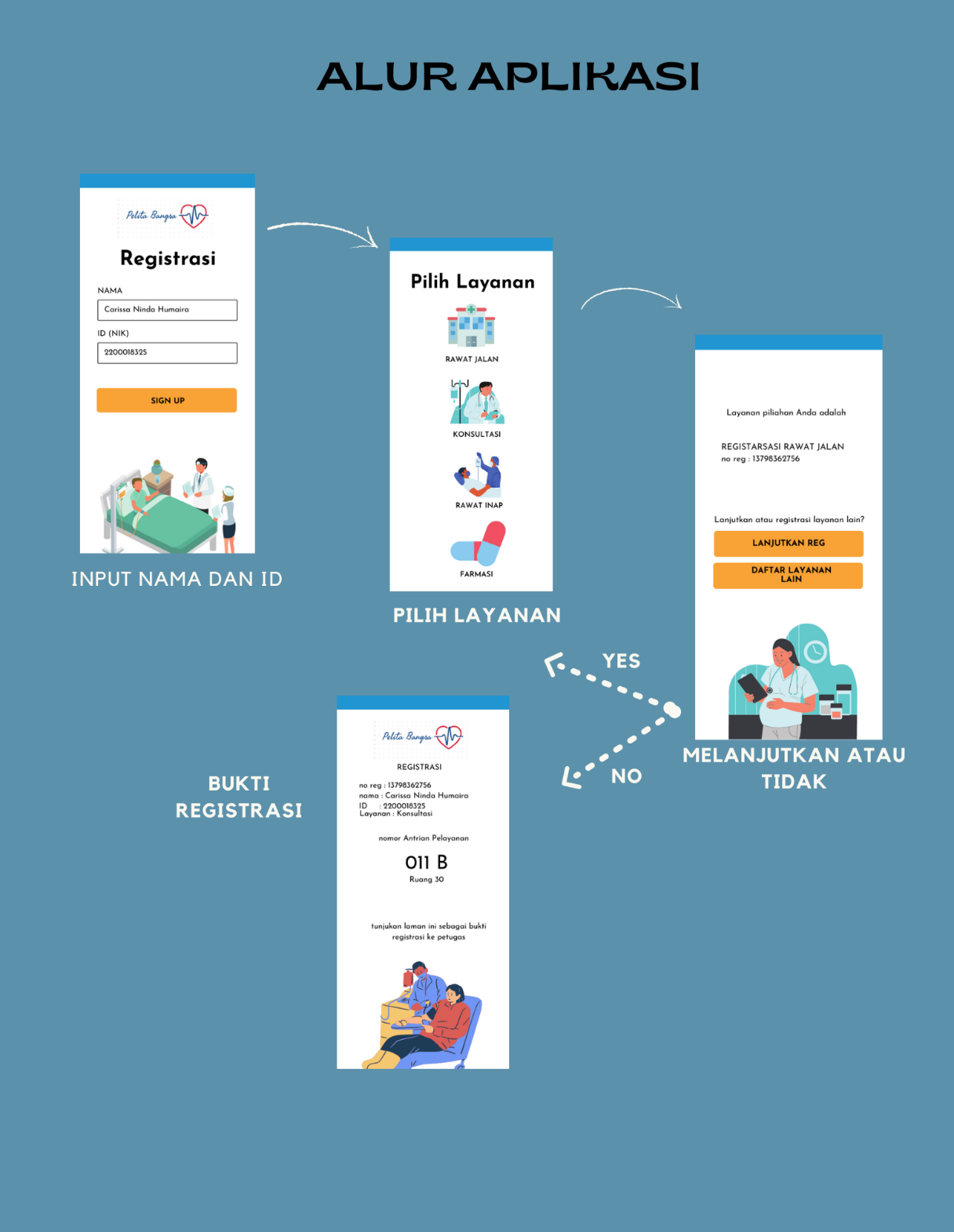
Pencatatan data untuk pelayanan rumah sakit kurang efisien dari sisi waktu, penumpukan kertas registrasi disaat Rumah sakit padat yang akan mengakibatkan tidak urutnya nomor pendaftaran sehigga pasien mendapat peyanan yang lambat, selain itu juga pendaftaran secara manual akan memakan biaya operasional yang lebih tinggi, namun hal tersebut dapat dikurangi dengan adanya kemajuan dan penggunaan teknologi komputer, Pemanfaatan teknologi komputer, pasien tidak harus datang ke Rumah Sakit untuk mendaftarkan diri, melainkan dapat mendaftar melalui perangkat seluler dan komputer yang dikirim ke server Rumah sakit selanjutnya diteruskan ke Administrator Rumah Sakit.

Tujuan dari pembuatan aplikasi ini adalah untuk mempermudah cara registrasi. Dan juga dapat mengurangi keterlambatan pelayanan sehingga pasien dapat mendapat penanganan lebih cepat dan tentunya biaya operasionalnya lebih kecil.

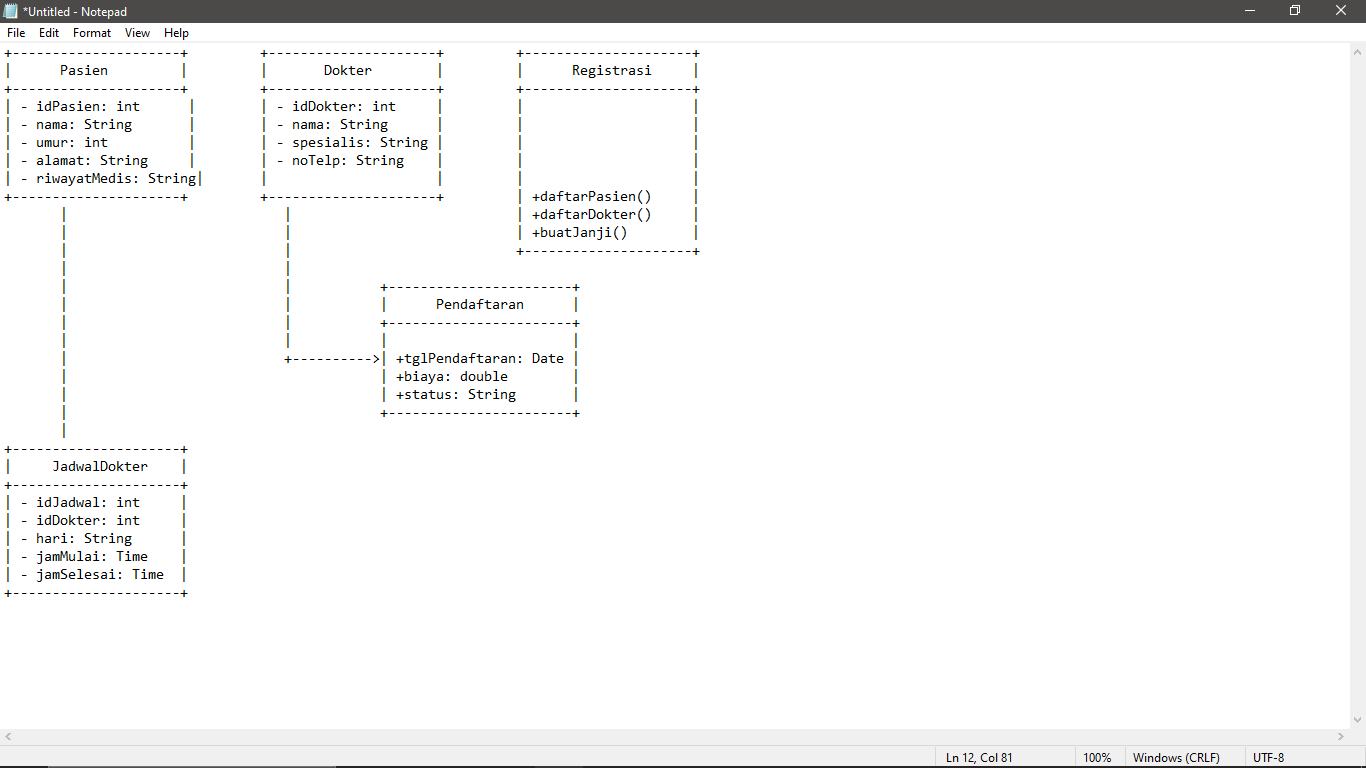
Jika me Run Program ini, Akan muncul jendela yang akan menamapilkan perintah untuk memasukkan nama, id di aplikasi yang telah di sediakan setelah meulis nama dan id maka akan muncul layanan yang ingin pasien pilih, lalu ketik no layanan yang ingin pasien daftarkan sesuai no yang ada di program.

* **ALUR DAN FLOWCHART**

****

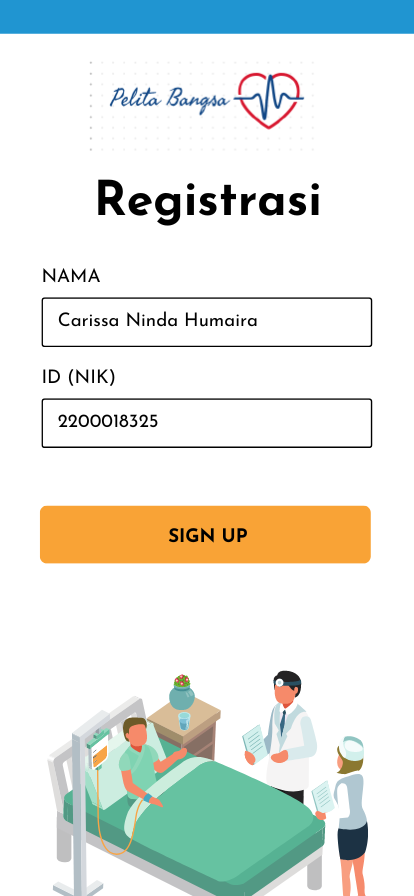


* **Diagram Class**



1. Kelas Pasien merepresentasikan data pasien seperti idPasien, nama, umur, alamat, dan riwayat medis.
2. Kelas Dokter merepresentasikan data dokter seperti idDokter, nama, spesialisasi, dan nomor telepon.
3. Kelas Registrasi memiliki metode untuk mendaftarkan pasien, dokter, dan membuat janji.
4. Kelas Pendaftaran merepresentasikan informasi pendaftaran pasien, termasuk tanggal pendaftaran, biaya, dan status.
5. Kelas JadwalDokter digunakan untuk menyimpan jadwal dokter, termasuk hari, jam mulai, dan jam selesai.

* Rancangan UI



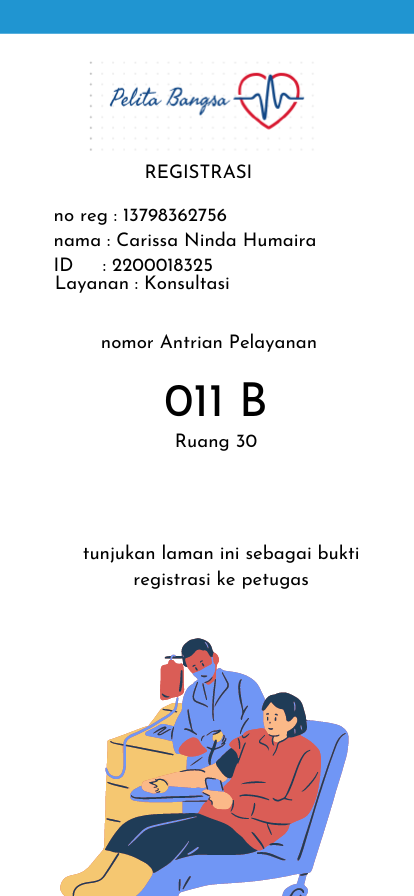
Tampilan aplikasi untuk Pengisian nama dan ID



Tampiolan aplikasi untuk pemilihan layanan

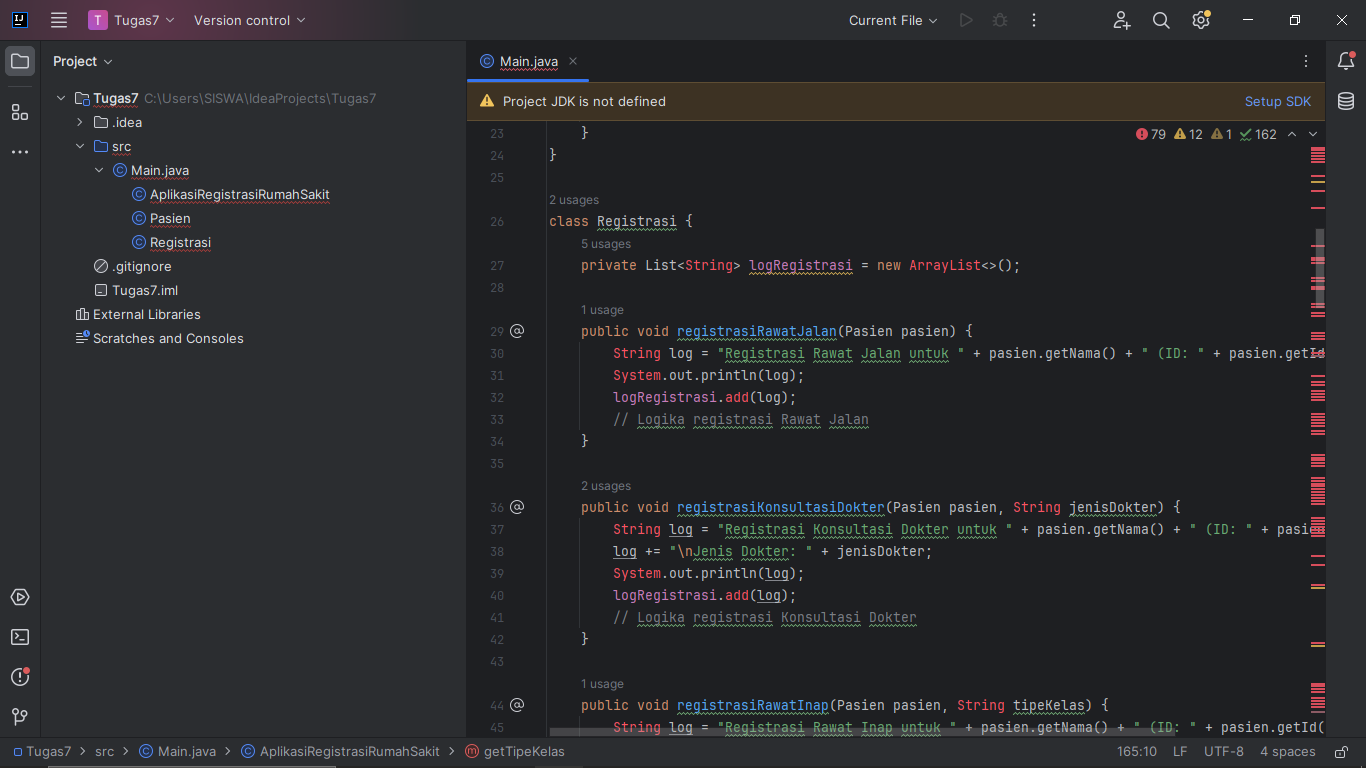
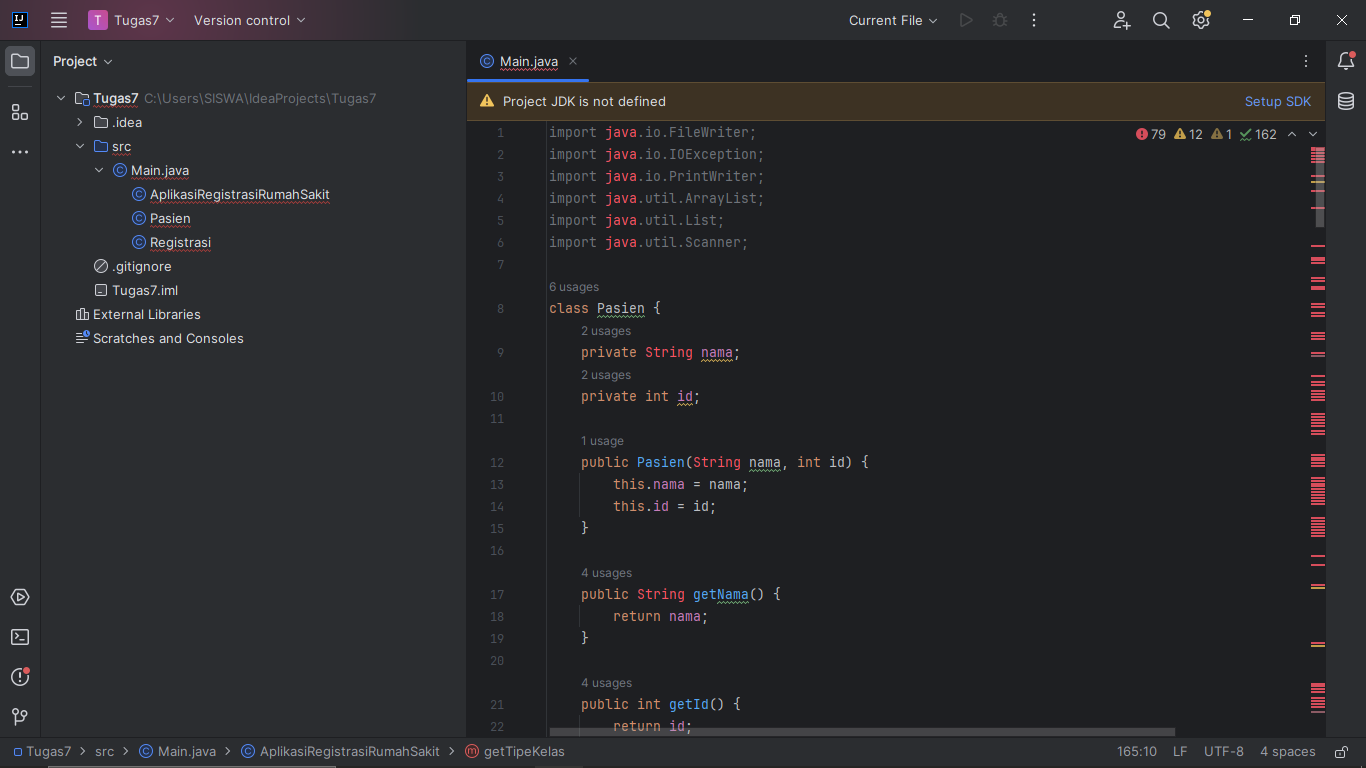


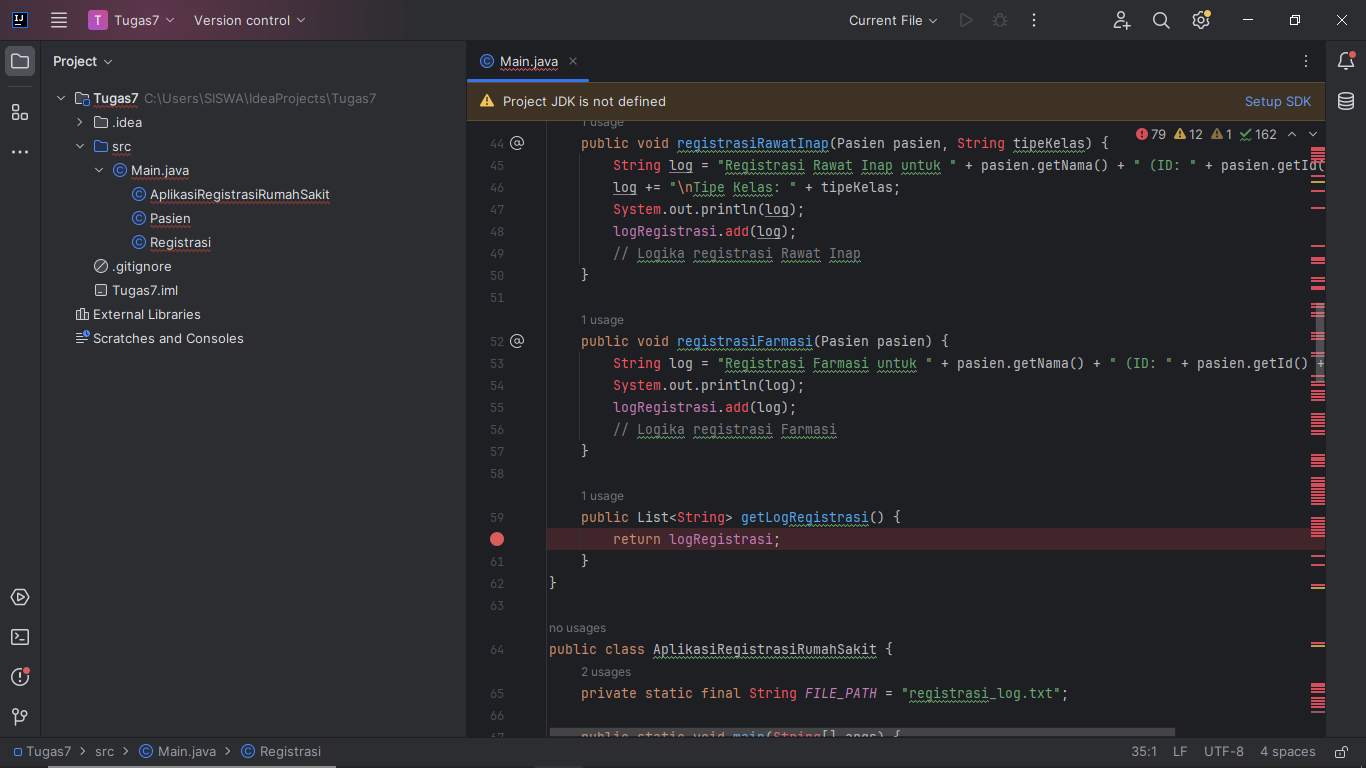
Tampilan aplikasi untuk pilihan melanjutkan atau memilih layanan lain

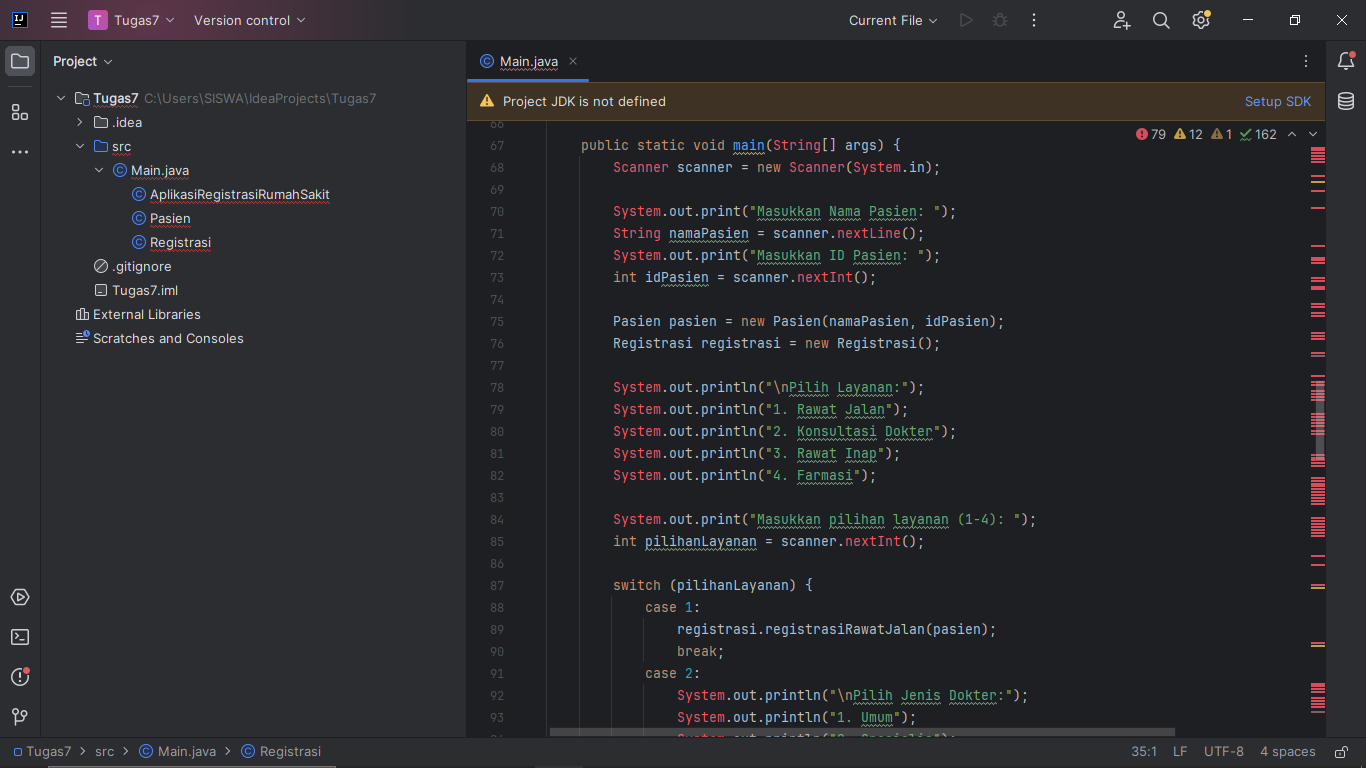


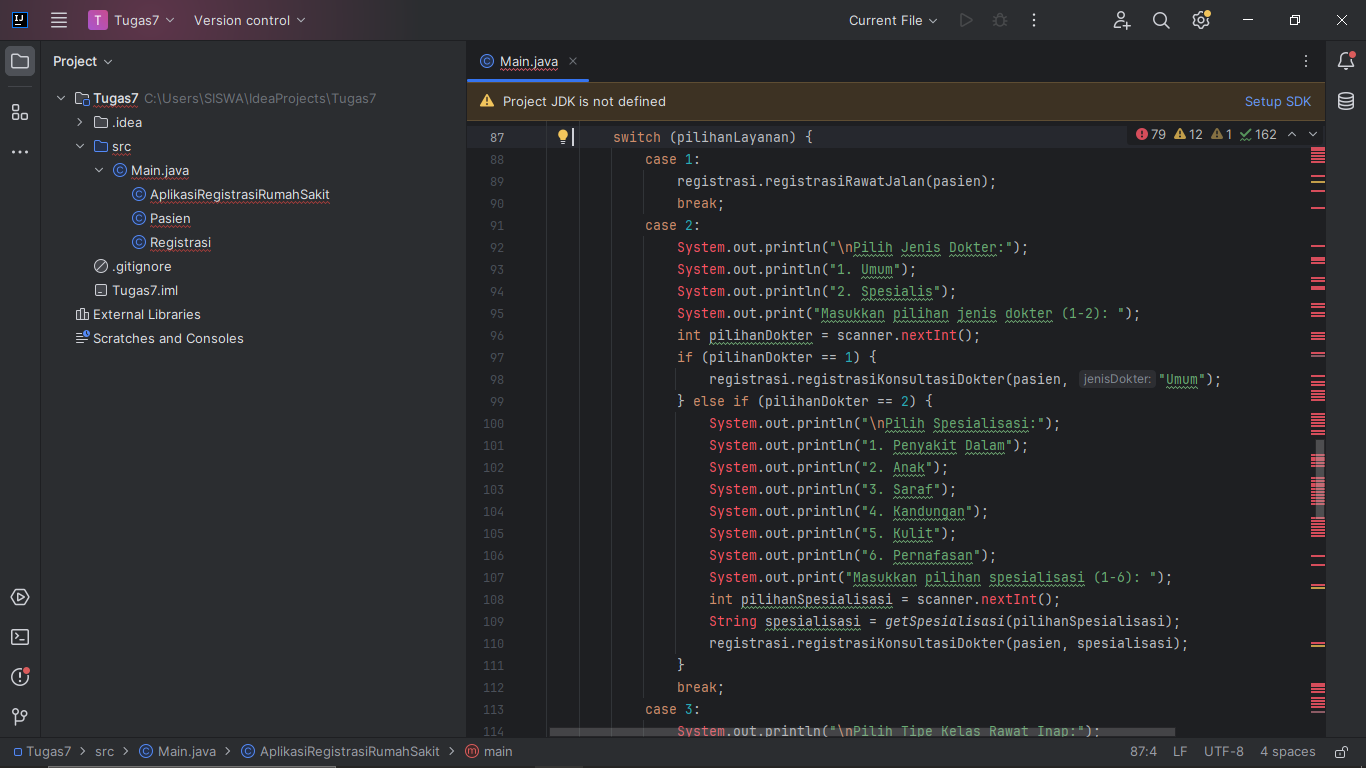
Tampilan aplikasi untuk Bukti Registrasi

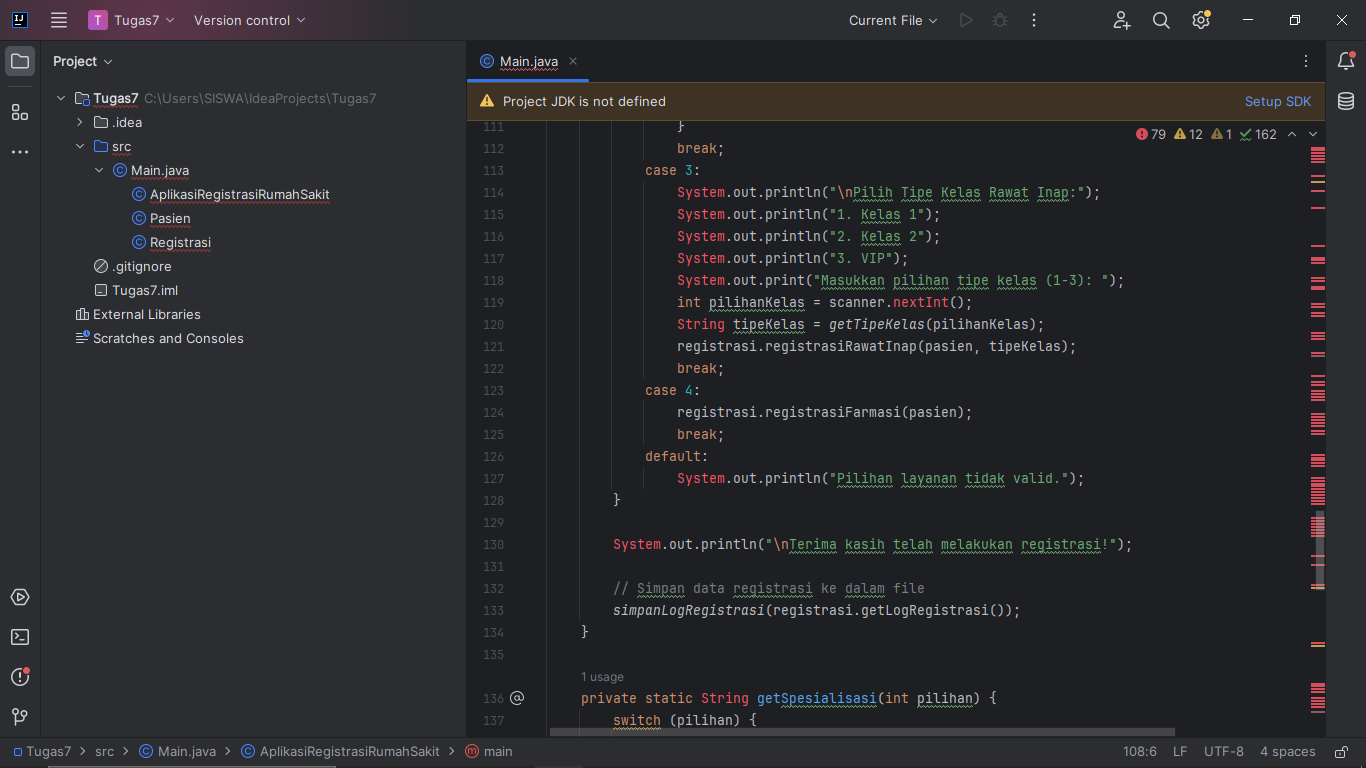
* **ScreenShoot kode kode program untuk menjalankan aplikasinya**

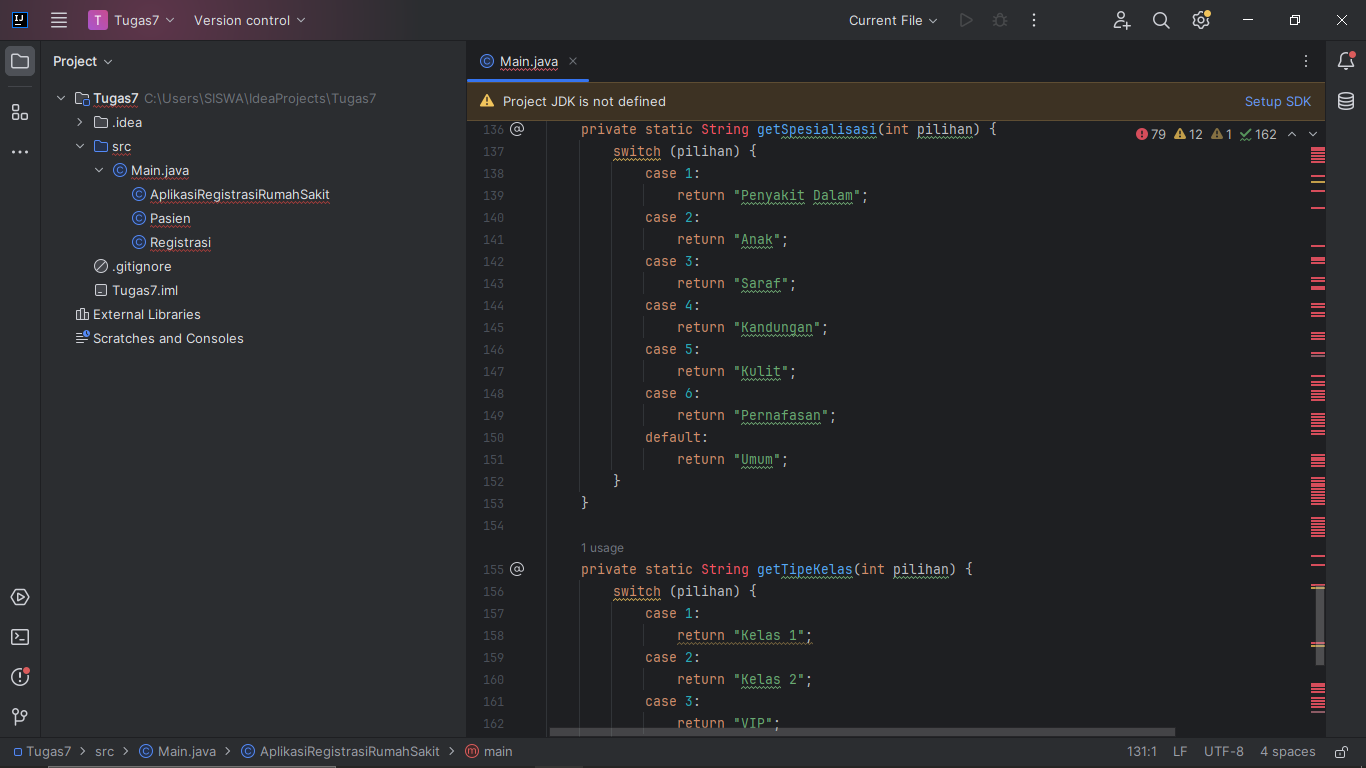


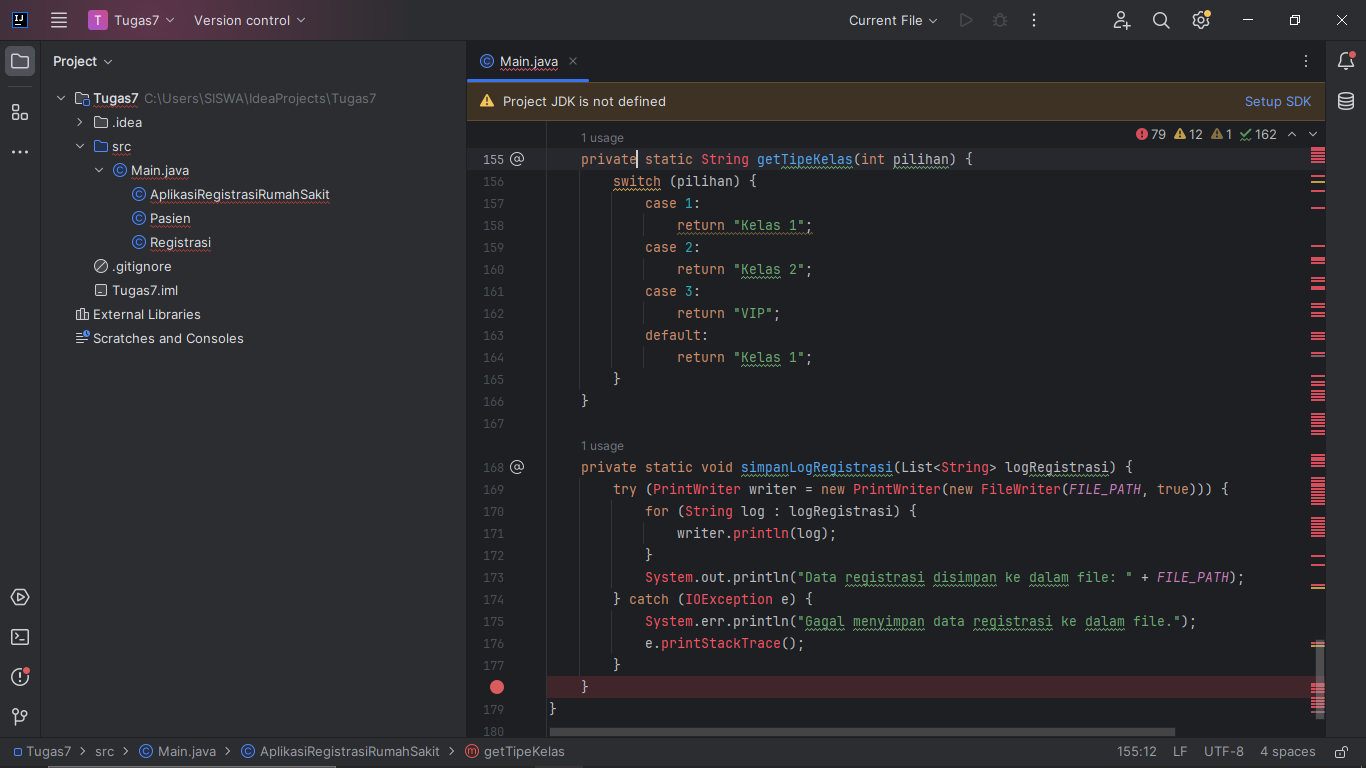












* **Penjelasan Koding**

Program di atas adalah sebuah aplikasi registrasi rumah sakit sederhana yang memungkinkan pengguna untuk mendaftar layanan seperti rawat jalan, konsultasi dokter, rawat inap, dan farmasi. Program ini dibuat dalam bahasa Java dan menggunakan konsep OOP (Object-Oriented Programming) dengan dua kelas utama, yaitu `Pasien` dan `Registrasi`.

1. Kelas Pasien:

- Kelas ini digunakan untuk merepresentasikan data pasien, seperti nama dan ID pasien.

- Konstruktor menerima dua parameter, yaitu nama dan ID, untuk menginisialisasi objek Pasien.

- Terdapat dua metode getter (`getNama` dan `getId`) untuk mendapatkan nilai nama dan ID pasien.

2. Kelas Registrasi:

- Kelas ini memiliki beberapa metode untuk melakukan registrasi berbagai layanan di rumah sakit.

- Setiap metode registrasi mencetak pesan ke layar dan menyimpan log registrasi dalam bentuk string di dalam list `logRegistrasi`.

- Terdapat metode `getLogRegistrasi` yang mengembalikan list log registrasi.

3. Kelas AplikasiRegistrasiRumahSakit:

- Kelas ini berisi metode `main` yang merupakan titik awal eksekusi program.

- Pada awalnya, program meminta pengguna memasukkan nama dan ID pasien.

- Objek `Pasien` dan `Registrasi` dibuat berdasarkan input pengguna.

- Pengguna kemudian diminta memilih layanan, dan program akan menangani pilihan tersebut menggunakan struktur kontrol `switch`.

- Bergantung pada pilihan, metode yang sesuai dari kelas `Registrasi` dipanggil untuk melakukan registrasi.

- Setelah registrasi, program menyimpan log registrasi ke dalam file teks menggunakan metode `simpanLogRegistrasi`.

4. Metode Tambahan (`getSpesialisasi`, `getTipeKelas`, `simpanLogRegistrasi`):

- `getSpesialisasi` dan `getTipeKelas` digunakan untuk mengonversi pilihan numerik pengguna menjadi string yang sesuai dengan jenis dokter atau tipe kelas.

- `simpanLogRegistrasi` digunakan untuk menyimpan log registrasi ke dalam file teks. Data ini disimpan dalam mode append, sehingga log sebelumnya tidak akan hilang.

* Fitur Tambahan:

- Setiap layanan mencetak informasi registrasi ke layar.

- Log registrasi disimpan dalam list dan dapat diakses melalui metode `getLogRegistrasi`.

- Data registrasi dapat disimpan ke dalam file teks (`registrasi\_log.txt`).

* **Kode program**

import java.io.FileWriter;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

class Pasien {

private String nama;

private int id;

public Pasien(String nama, int id) {

this.nama = nama;

this.id = id;

}

public String getNama() {

return nama;

}

public int getId() {

return id;

}

}

class Registrasi {

private List<String> logRegistrasi = new ArrayList<>();

public void registrasiRawatJalan(Pasien pasien) {

String log = "Registrasi Rawat Jalan untuk " + pasien.getNama() + " (ID: " + pasien.getId() + ")";

System.out.println(log);

logRegistrasi.add(log);

// Logika registrasi Rawat Jalan

}

public void registrasiKonsultasiDokter(Pasien pasien, String jenisDokter) {

String log = "Registrasi Konsultasi Dokter untuk " + pasien.getNama() + " (ID: " + pasien.getId() + ")";

log += "\nJenis Dokter: " + jenisDokter;

System.out.println(log);

logRegistrasi.add(log);

// Logika registrasi Konsultasi Dokter

}

public void registrasiRawatInap(Pasien pasien, String tipeKelas) {

String log = "Registrasi Rawat Inap untuk " + pasien.getNama() + " (ID: " + pasien.getId() + ")";

log += "\nTipe Kelas: " + tipeKelas;

System.out.println(log);

logRegistrasi.add(log);

// Logika registrasi Rawat Inap

}

public void registrasiFarmasi(Pasien pasien) {

String log = "Registrasi Farmasi untuk " + pasien.getNama() + " (ID: " + pasien.getId() + ")";

System.out.println(log);

logRegistrasi.add(log);

// Logika registrasi Farmasi

}

public List<String> getLogRegistrasi() {

return logRegistrasi;

}

}

public class AplikasiRegistrasiRumahSakit {

private static final String FILE\_PATH = "registrasi\_log.txt";

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Masukkan Nama Pasien: ");

String namaPasien = scanner.nextLine();

System.out.print("Masukkan ID Pasien: ");

int idPasien = scanner.nextInt();

Pasien pasien = new Pasien(namaPasien, idPasien);

Registrasi registrasi = new Registrasi();

System.out.println("\nPilih Layanan:");

System.out.println("1. Rawat Jalan");

System.out.println("2. Konsultasi Dokter");

System.out.println("3. Rawat Inap");

System.out.println("4. Farmasi");

System.out.print("Masukkan pilihan layanan (1-4): ");

int pilihanLayanan = scanner.nextInt();

switch (pilihanLayanan) {

case 1:

registrasi.registrasiRawatJalan(pasien);

break;

case 2:

System.out.println("\nPilih Jenis Dokter:");

System.out.println("1. Umum");

System.out.println("2. Spesialis");

System.out.print("Masukkan pilihan jenis dokter (1-2): ");

int pilihanDokter = scanner.nextInt();

if (pilihanDokter == 1) {

registrasi.registrasiKonsultasiDokter(pasien, "Umum");

} else if (pilihanDokter == 2) {

System.out.println("\nPilih Spesialisasi:");

System.out.println("1. Penyakit Dalam");

System.out.println("2. Anak");

System.out.println("3. Saraf");

System.out.println("4. Kandungan");

System.out.println("5. Kulit");

System.out.println("6. Pernafasan");

System.out.print("Masukkan pilihan spesialisasi (1-6): ");

int pilihanSpesialisasi = scanner.nextInt();

String spesialisasi = getSpesialisasi(pilihanSpesialisasi);

registrasi.registrasiKonsultasiDokter(pasien, spesialisasi);

}

break;

case 3:

System.out.println("\nPilih Tipe Kelas Rawat Inap:");

System.out.println("1. Kelas 1");

System.out.println("2. Kelas 2");

System.out.println("3. VIP");

System.out.print("Masukkan pilihan tipe kelas (1-3): ");

int pilihanKelas = scanner.nextInt();

String tipeKelas = getTipeKelas(pilihanKelas);

registrasi.registrasiRawatInap(pasien, tipeKelas);

break;

case 4:

registrasi.registrasiFarmasi(pasien);

break;

default:

System.out.println("Pilihan layanan tidak valid.");

}

System.out.println("\nTerima kasih telah melakukan registrasi!");

// Simpan data registrasi ke dalam file

simpanLogRegistrasi(registrasi.getLogRegistrasi());

}

private static String getSpesialisasi(int pilihan) {

switch (pilihan) {

case 1:

return "Penyakit Dalam";

case 2:

return "Anak";

case 3:

return "Saraf";

case 4:

return "Kandungan";

case 5:

return "Kulit";

case 6:

return "Pernafasan";

default:

return "Umum";

}

}

private static String getTipeKelas(int pilihan) {

switch (pilihan) {

case 1:

return "Kelas 1";

case 2:

return "Kelas 2";

case 3:

return "VIP";

default:

return "Kelas 1";

}

}

private static void simpanLogRegistrasi(List<String> logRegistrasi) {

try (PrintWriter writer = new PrintWriter(new FileWriter(FILE\_PATH, true))) {

for (String log : logRegistrasi) {

writer.println(log);

}

System.out.println("Data registrasi disimpan ke dalam file: " + FILE\_PATH);

} catch (IOException e) {

System.err.println("Gagal menyimpan data registrasi ke dalam file.");

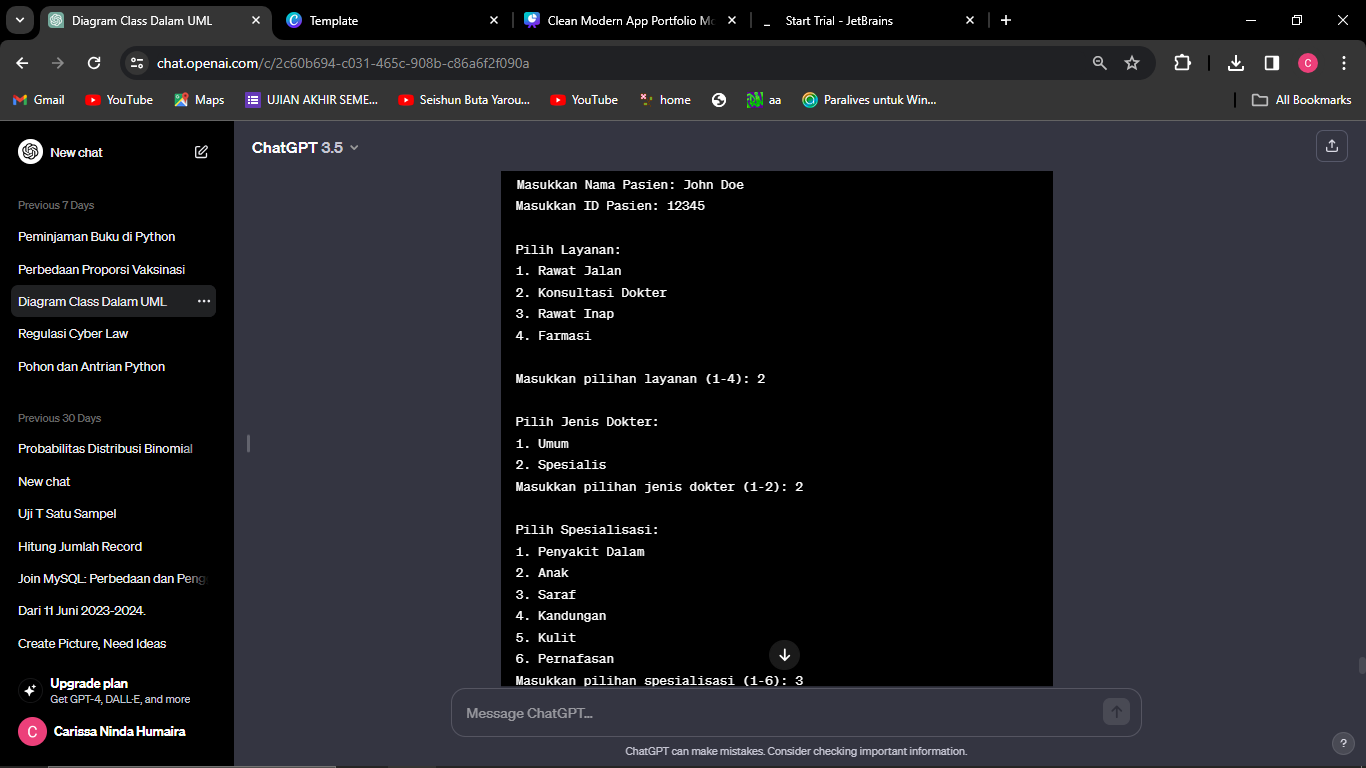
e.printStackTrace();

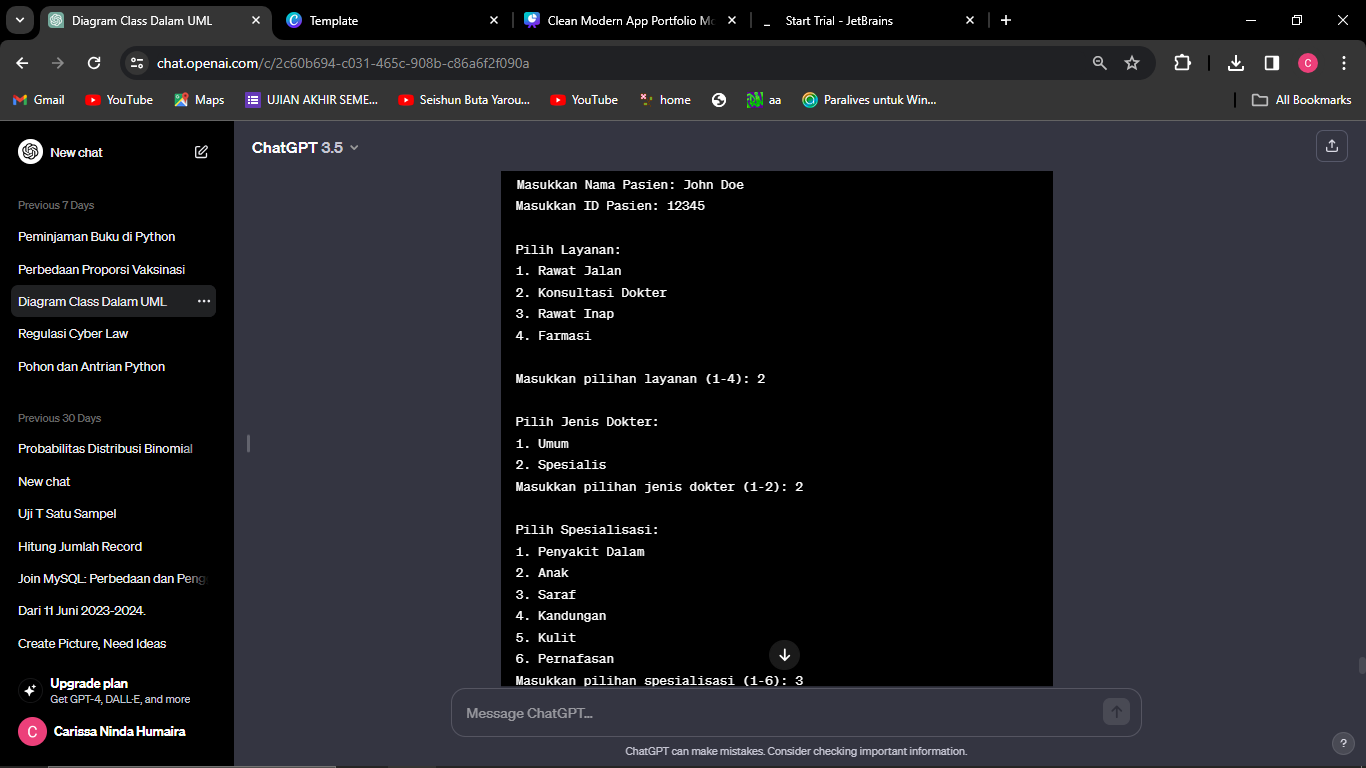
}

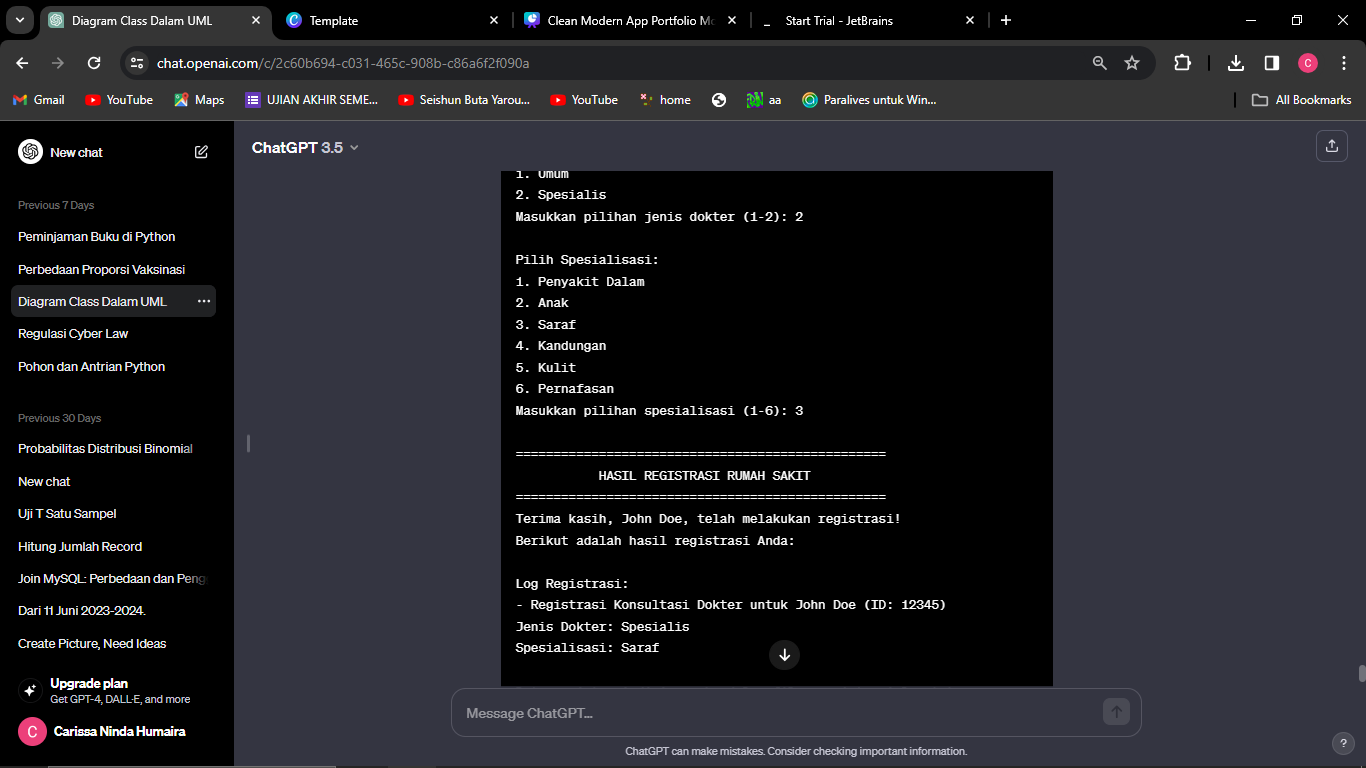
}

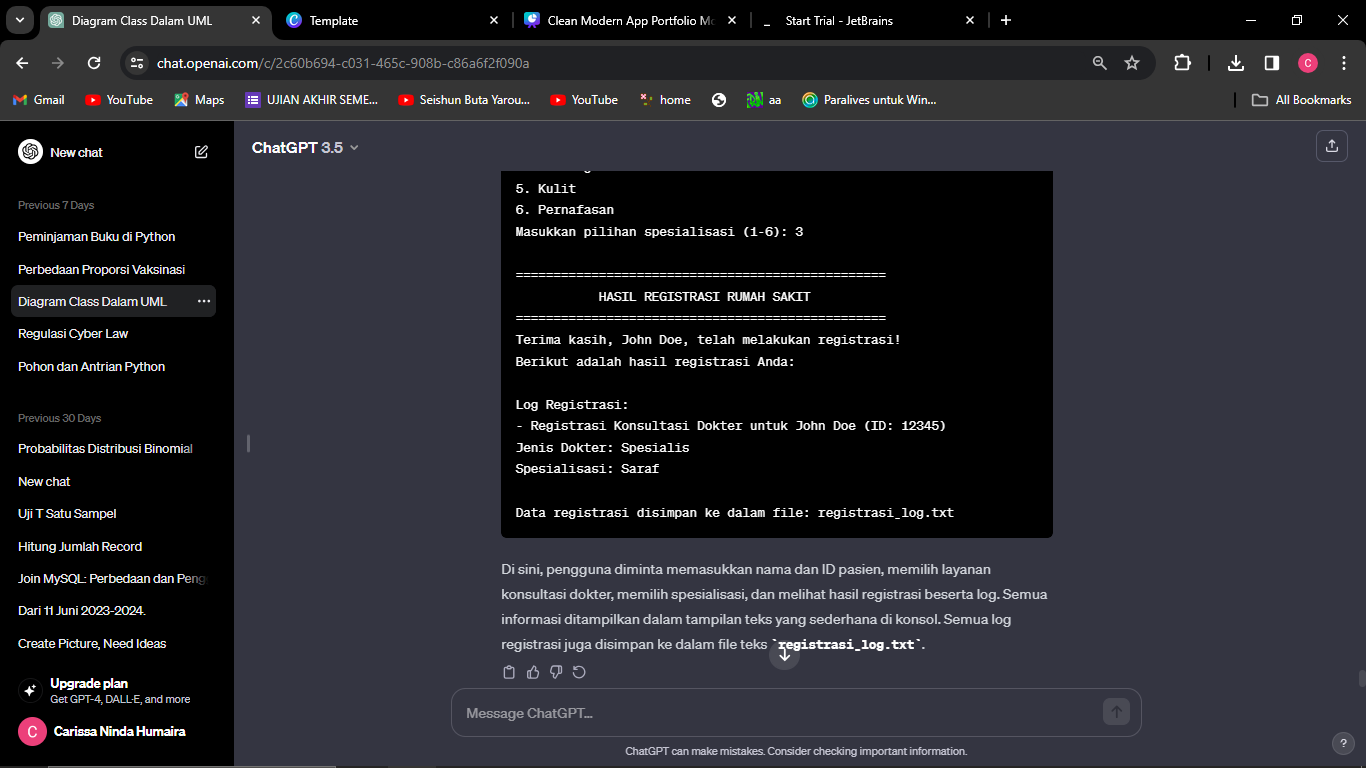
}

* **ScreenShot tampilan luar program**









**Penjelasan**

Dari Uji coba yang saya lakukan, saya menggunakan suasu contoh kasus Pasien yang ingin mendaftar layanan dokter spesialis

1. Masukkan Nama Pasien: John Doe

Pengguna diminta untuk memasukkan nama pasien, dan dalam contoh ini, pengguna memasukkan nama "John Doe".

1. Masukkan ID Pasien: 12345

Pengguna diminta untuk memasukkan ID pasien, dan dalam contoh ini, pengguna memasukkan ID "12345".

1. Pilih Layanan: 2 (Konsultasi Dokter)

Pengguna diminta untuk memilih layanan, dan dalam contoh ini, pengguna memilih layanan konsultasi dokter (pilihan 2).

1. Pilih Jenis Dokter: 2 (Spesialis)

Karena pengguna memilih layanan konsultasi dokter, program kemudian meminta pengguna untuk memilih jenis dokter. Dalam contoh ini, pengguna memilih spesialis (pilihan 2).

1. Pilih Spesialisasi: 3 (Saraf)

Jika pengguna memilih spesialis, program kemudian meminta pengguna untuk memilih spesialisasi dokter. Dalam contoh ini, pengguna memilih spesialisasi saraf (pilihan 3).

1. Hasil Registrasi dan Log Registrasi

Program mencetak hasil registrasi ke layar, termasuk pesan terima kasih dan hasil registrasi seperti nama, ID, jenis dokter, dan spesialisasi.

Program juga mencetak log registrasi ke layar berisi informasi registrasi seperti jenis layanan, nama, ID, dan keterangan tambahan.

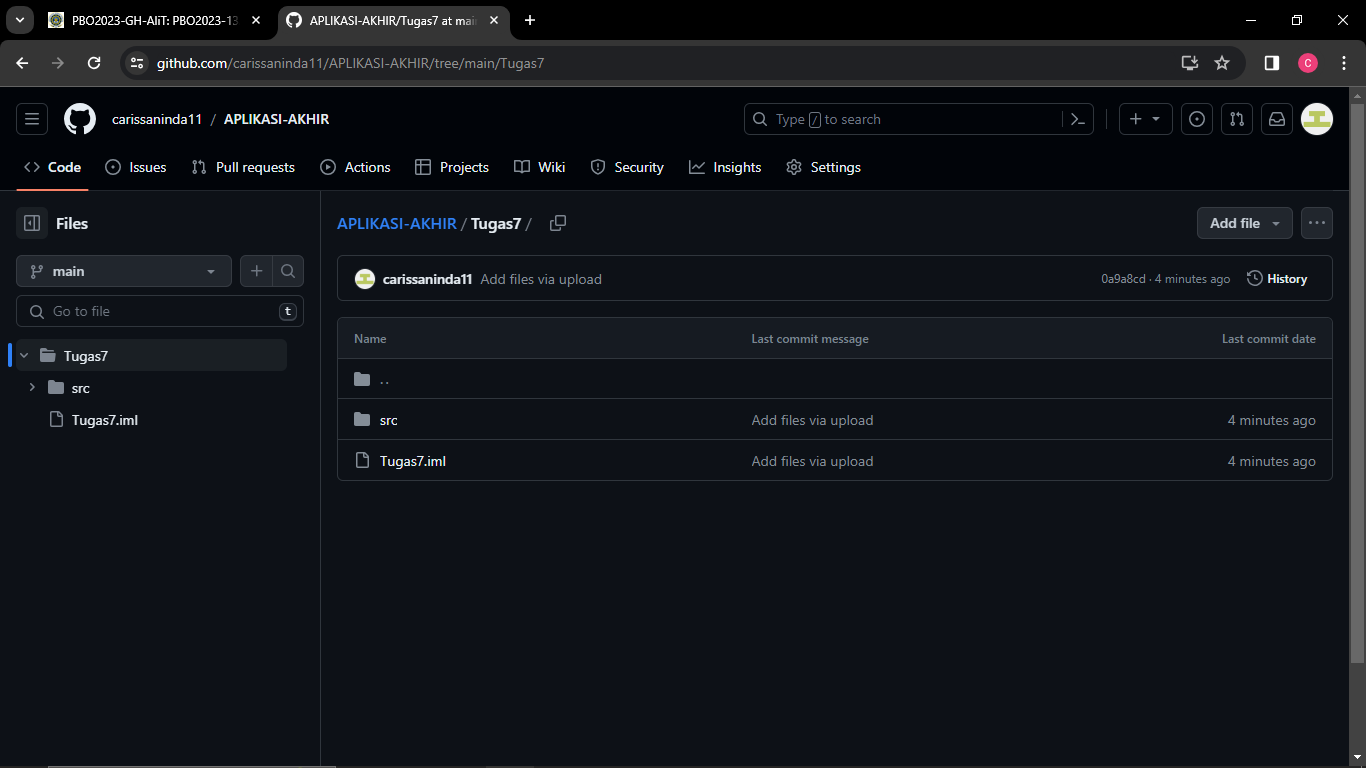
Contoh di atas menunjukkan hasil registrasi untuk konsultasi dokter spesialis saraf.

1. Simpan Log Registrasi ke dalam File: registrasi\_log.txt

Program menyimpan semua log registrasi ke dalam file teks registrasi\_log.txt untuk dicatat dan dapat dilihat kembali nanti.

Hasil ini memberikan gambaran simulasi dari interaksi pengguna dengan program registrasi rumah sakit. Semua langkah-langkah ini dapat dijalankan dari awal hingga akhir saat Anda menjalankan program dalam lingkungan Java yang sesuai.

* **Screenshot tampilan unggahan daftar file di halaman projek Gitlab**



* **Kesimpulan**

Melalui pengembangan program proyek akhir registrasi rumah sakit menggunakan Java, saya memperoleh pemahaman yang kokoh tentang konsep-konsep dasar pemrograman berorientasi objek (OOP) dan implementasi praktisnya. Proyek ini memperluas pengetahuan saya dalam interaksi dengan pengguna melalui input dan output, pengambilan keputusan, dan penanganan file. Penerapan struktur kontrol, manipulasi struktur data, dan pemrosesan string dalam konteks nyata memperkaya pengalaman belajar saya, dan kemampuan untuk menyimpan dan membaca informasi dari file teks memberikan wawasan tentang operasi dasar file I/O.

Dengan menyelesaikan proyek ini, harapan saya adalah untuk terus memperdalam pemahaman saya dalam pengembangan aplikasi Java serta mengembangkan kemampuan analisis dan desain yang lebih matang. Saya berambisi untuk terlibat dalam proyek-proyek yang lebih kompleks, menggali lebih dalam konsep keamanan perangkat lunak, dan menerapkan inovasi serta kreativitas dalam pengembangan solusi teknologi yang memberikan dampak positif. Proyek ini menjadi batu loncatan untuk terus belajar, berkolaborasi dengan rekan tim, dan membangun keterampilan yang diperlukan untuk mengatasi tantangan nyata dalam dunia pengembangan perangkat lunak.