

Penjelasan Integrasi Front End & Back End

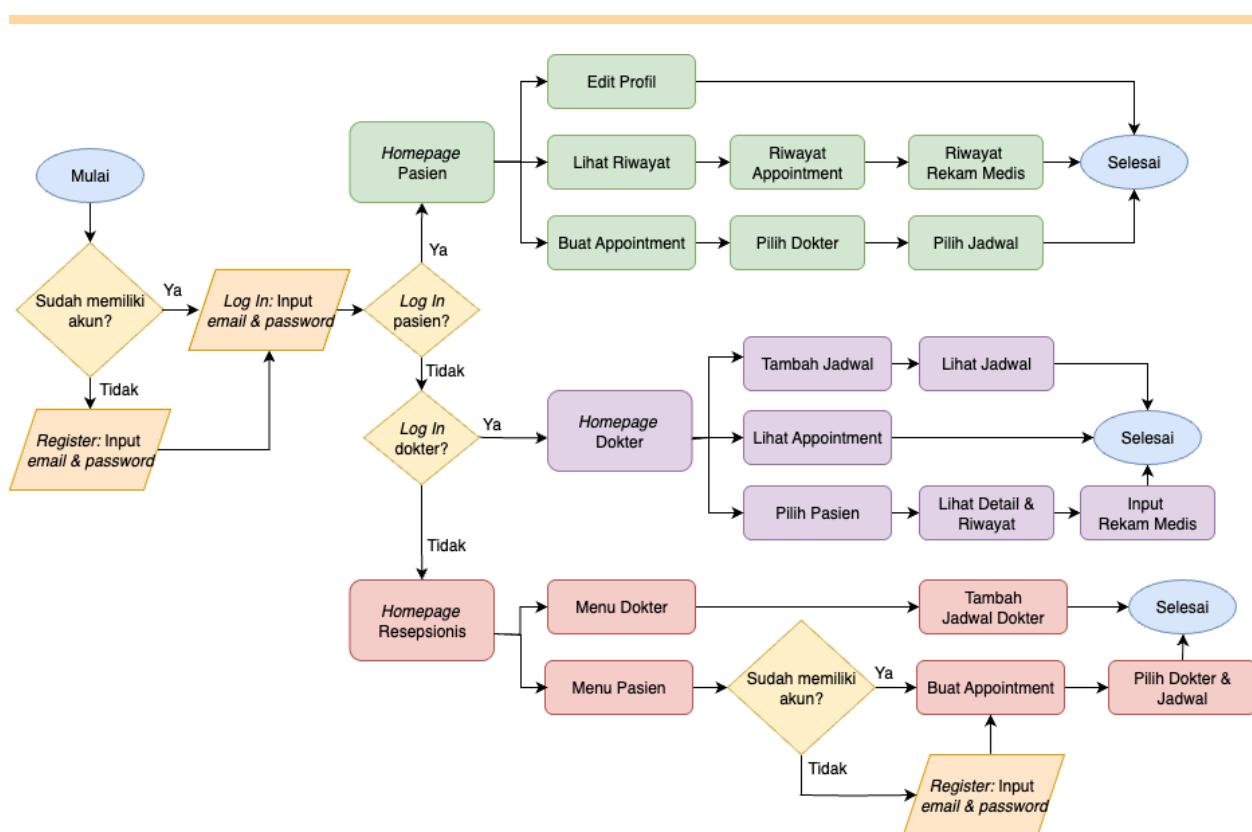
Week 4 - Application Integration

Kelas : KOM-A

Anggota : Aliya Khairun Nisa (24/543832/PA/23111)

Farsya Nabila Tori (24/543855/PA/23113)

Gracella Wiendy Koesnadi (24/541739/PA/22991)



Gambar 1. Diagram Alir (Flowchart) Penggunaan Website

Front End & Back End Connection

Mekanisme koneksi dan interaksi antara database Relasional (MySQL) dengan web frontend (HTML/Jinja) pada proyek ini diimplementasikan melalui framework Python Flask. Proses ini mengikuti arsitektur Model-View-Controller (MVC) atau varian serupa yang memisahkan logika backend dari presentasi frontend. Selain itu, database di-deploy ke

Railway agar semua perangkat yang mengakses aplikasi ini menggunakan database yang sama. Berikut adalah penjelasan rinci mengenai mekanisme koneksi dan interaksi data:

1) Lapisan Koneksi Database (Backend)

Koneksi database diatur sepenuhnya dalam file app.py yang bertindak sebagai server aplikasi (Controller dan Model).

- **Konfigurasi dan Inisialisasi Koneksi:** Aplikasi menggunakan library flask_mysqldb yang berfungsi sebagai adaptor antara Flask dan MySQL, serta pymysql sebagai driver koneksi.
 - Kredensial sensitif database (Host, User, Password, DB Name, Port) tidak ditulis langsung dalam kode, melainkan diambil dari variabel .env (menggunakan os.getenv dan dotenv).
 - Detail koneksi diatur dalam konfigurasi aplikasi Flask (app.config).
 - Objek MySQL(app) diinisialisasi sehingga dapat membuat koneksi database yang siap digunakan oleh seluruh aplikasi.

Python

```
# app.py: Inisialisasi dan Konfigurasi

import pymysql
pymysql.install_as_MySQLdb()
from flask_mysqldb import MySQL
from dotenv import load_dotenv
import os

load_dotenv()

# Mengambil kredensial dari environment variables (keamanan)
app.config['MYSQL_HOST'] = os.getenv('MYSQL_HOST')
app.config['MYSQL_USER'] = os.getenv('MYSQL_USER')
app.config['MYSQL_PASSWORD'] = os.getenv('MYSQL_PASSWORD')
app.config['MYSQL_DB'] = os.getenv('MYSQL_DB')
```

```
# Inisialisasi objek MySQL
mysql = MySQL(app)
```

- **Eksekusi Perintah SQL:** Interaksi data dilakukan menggunakan objek Cursor yang diperoleh dari objek koneksi:
 - Membuka Kursor: Untuk setiap operasi database (SELECT, INSERT, UPDATE, DELETE), kursor dibuka dengan "cur = mysql.connection.cursor()".
 - Eksekusi Query: Kursor digunakan untuk mengeksekusi perintah SQL melalui cur.execute("QUERY", (parameter,)). Semua input dari pengguna (seperti email atau password pada login atau data formulir) selalu diteruskan sebagai parameter terpisah (%s) untuk mencegah serangan SQL Injection.
 - Pengambilan Hasil:
 - cur.fetchone(): Digunakan untuk mengambil satu baris hasil (misalnya saat login atau mengambil detail profil).
 - cur.fetchall(): Digunakan untuk mengambil semua baris hasil (misalnya saat menampilkan daftar dokter atau riwayat janji temu).
 - Commit Perubahan: Untuk operasi modifikasi data (INSERT, UPDATE, DELETE), perubahan permanen dilakukan dengan mysql.connection.commit().
 - Menutup Kursor: Kursor selalu ditutup dengan cur.close() setelah operasi selesai untuk melepaskan sumber daya database.

2) Lapisan Interaksi Data (Frontend)

Interaksi data antara backend (Python) dan frontend (HTML) difasilitasi oleh Flask Routes dan template engine Jinja2.

- **Alur Data (Read Operation)**

- Permintaan Klien (Frontend): Pengguna mengakses URL tertentu (misalnya /profile atau /appointmentHistory).
- Pemrosesan Route (Backend): Fungsi route Flask terkait dieksekusi. Fungsi ini melakukan otentikasi sesi (if 'email' not in session:), menjalankan query SQL kompleks (misalnya join tabel Appointment, Pasien, Dokter, dan Rekam Medis) menggunakan kursor, dan mengambil hasil (history_data = cur.fetchall()).
- Rendering Template: Hasil data (berbentuk list atau dictionary) dilewatkan ke template HTML menggunakan render_template(). Contoh: return render_template('profile.html', user=user_info, phones=phone_list).
- Template Engine (Frontend): Jinja2 menerima variabel data dan mengulanginya (loop) untuk membuat HTML dinamis (misalnya {% for app in appointments %}).

- **Alur Data (Write/Update Operation)**

- Permintaan Klien (Frontend): Pengguna mengisi formulir HTML (misalnya /editProfile) dan menekan tombol submit (metode POST).
- Pengambilan Data Formulir (Backend): Fungsi route Flask menangkap data yang dikirim melalui request.form['nama_kolom'].
- Validasi/Pemrosesan: Data diproses (misalnya, hashing password saat registrasi atau validasi input telepon).
- Eksekusi Query SQL: Perintah INSERT atau UPDATE dieksekusi menggunakan kursor dan data yang diterima dari formulir.
- Pesan Umpam Balik: Aplikasi menggunakan flash() untuk menyimpan pesan status (sukses atau gagal) di sesi, yang kemudian ditampilkan di layout halaman berikutnya.
- Redirect: Pengguna diarahkan ulang ke halaman lain (misalnya return redirect(url_for('profile'))) untuk mencegah pengiriman formulir ganda.

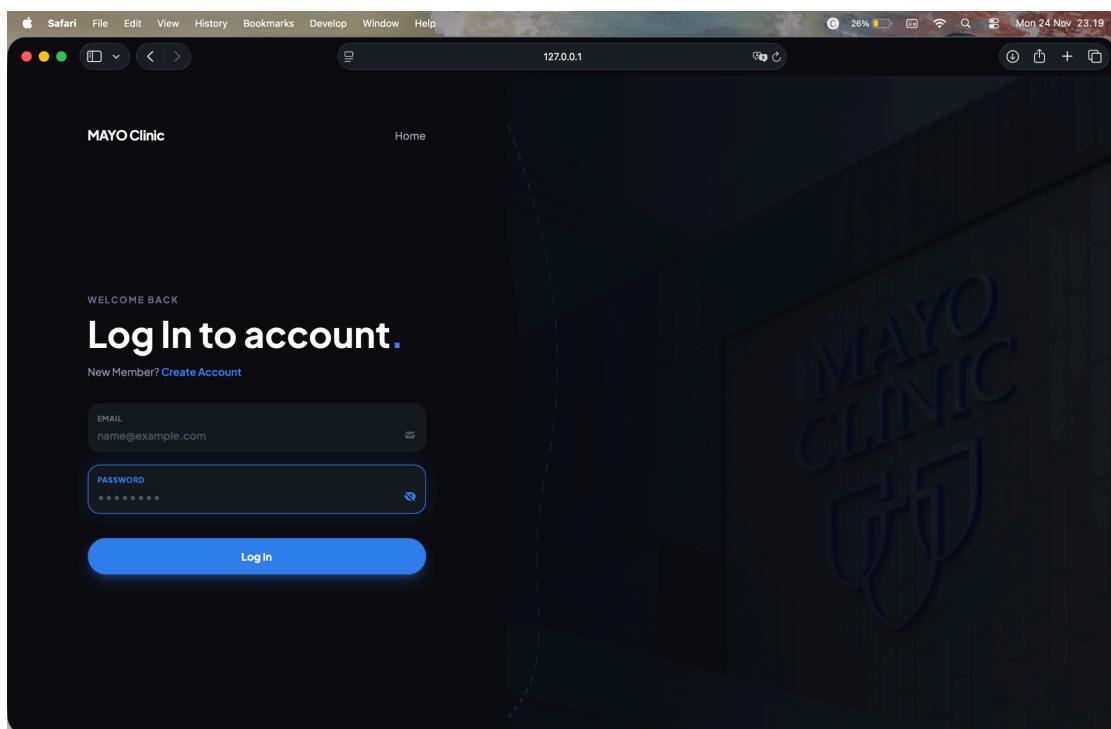
Secara keseluruhan, proyek ini menggunakan pendekatan server-side rendering yang memisahkan tanggung jawab database (MySQL), logika aplikasi (Flask/Python),

dan presentasi data (HTML/Jinja2), memastikan alur data yang aman dan terstruktur.

Snapshot Implementasi

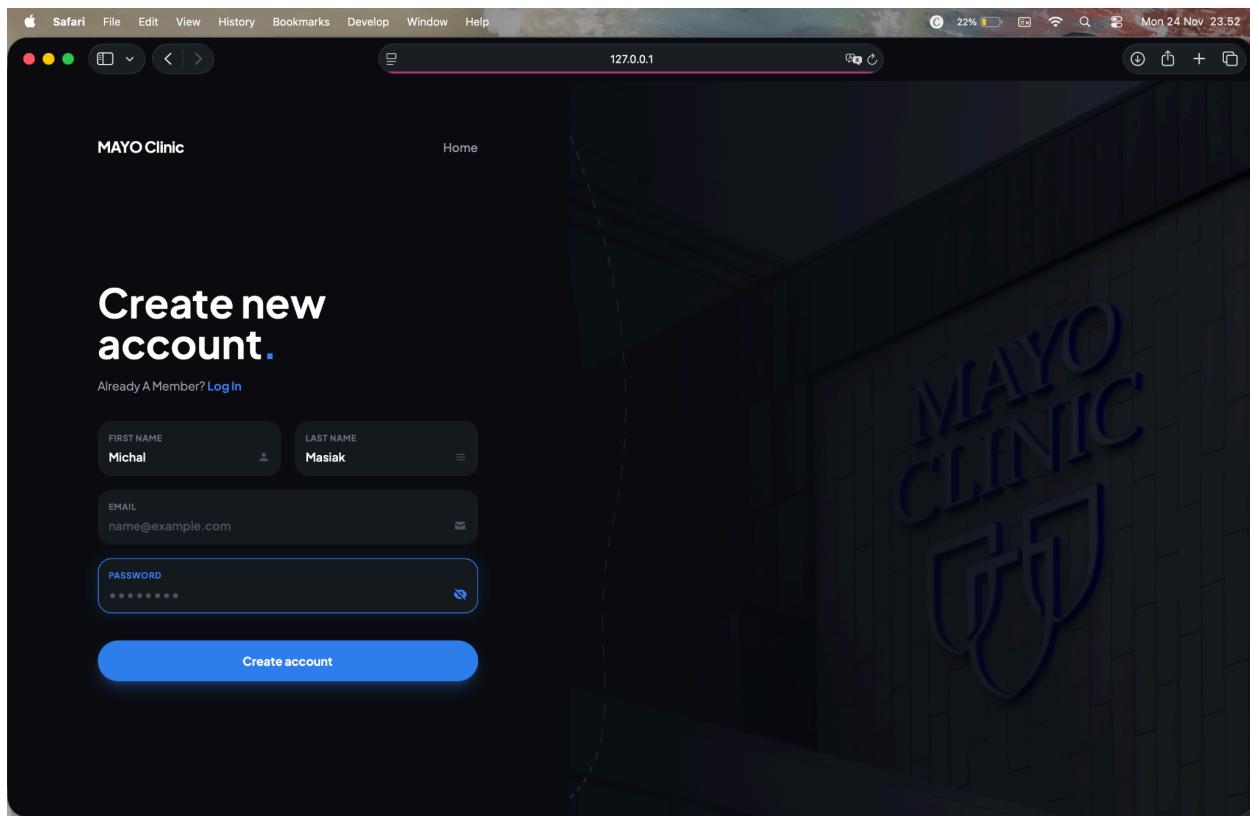
Universal

Page login ini akan diakses oleh pasien, dokter, dan resepsionis. Formulir login terdiri dari dua field input utama yaitu Email, di mana pengguna memasukkan alamat email mereka, yang berfungsi sebagai username untuk autentikasi sistem. Field kedua adalah Password, yang memiliki fitur tombol toggle mata di samping field yang memungkinkan pengguna menampilkan atau menyembunyikan karakter password (fungsi `togglePassword()`). Formulir ini mengirimkan data menggunakan metode POST ke route `/login` di backend, di mana server akan memproses autentikasi. Pesan sistem seperti notifikasi login berhasil atau pesan error "Invalid email or password", ditampilkan di atas formulir melalui mekanisme flashed messages Flask. Setelah mengisi kedua field, pengguna menekan tombol Log In berwarna biru untuk memproses permintaan masuk. Background page ini gradien dari warna ke gambar (mohon tingkatkan kecerahan layar agar dapat melihat gambar dengan jelas).

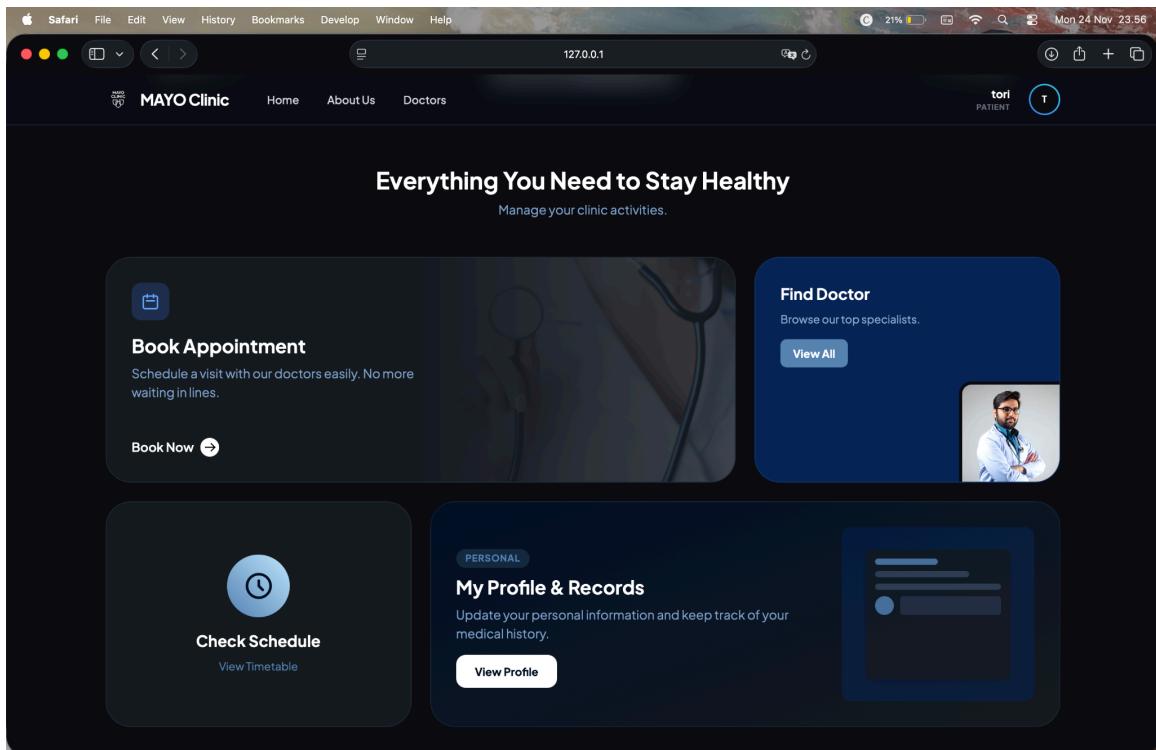
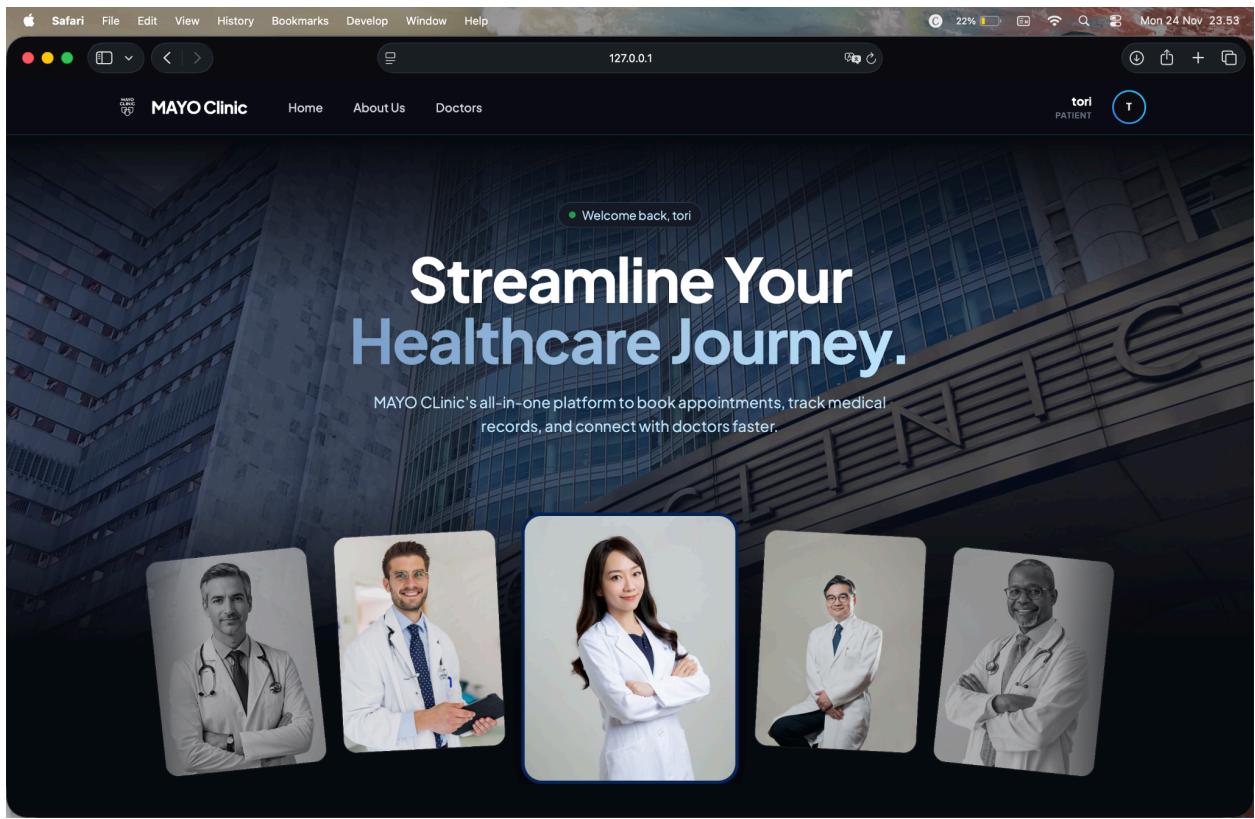


Pasien

Pasien akan register di halaman ini sebelum membuat janji temu. Form register berisi email dan password. yang memiliki fitur tombol toggle mata di samping field yang memungkinkan pengguna menampilkan atau menyembunyikan karakter password (fungsi `togglePassword()`). Password kemudian akan dihash sebelum dimasukkan ke database



Gambar di bawah adalah homepage pasien yang berisi 5 container. Container pertama sebagai pembuka, container kedua berisi utilities/fitur yang ada pada website, container ketiga adalah penjelasan singkat mengenai klinik. Lalu container keempat berisi daftar dokter beserta fotonya dan container terakhir berisi informasi tentang kontak klinik.



Safari File Edit View History Bookmarks Develop Window Help 21% Mon 24 Nov 23:57

127.0.0.1 tori PATIENT

MAYO Clinic Home About Us Doctors View Profile

About MAYO Clinic

MAYO Clinic is a world-leading not-for-profit health care system focused on complex care, research and medical education. Founded in 1864 in Rochester, Minnesota, the clinic is known for its highly specialized services and is consistently ranked as one of the best hospitals in the United States.

Opening Hours
Mon - Fri: 08:00 - 20:00
Sat: 09:00 - 17:00

Location
13400 E. Shea Blvd, Scottsdale, AZ 85259
Rochester, Minnesota, United States



Our Doctors
Meet our experienced medical team

Safari File Edit View History Bookmarks Develop Window Help 21% Mon 24 Nov 23:57

127.0.0.1 tori PATIENT

MAYO Clinic Home About Us Doctors

Our Doctors

Meet our experienced medical team



Dr. Rafathin Ardian
Active



Dr. James Wilson
Active



Dr. Emily Carter
Active



Dr. Michael Brown
Active

Get in Touch

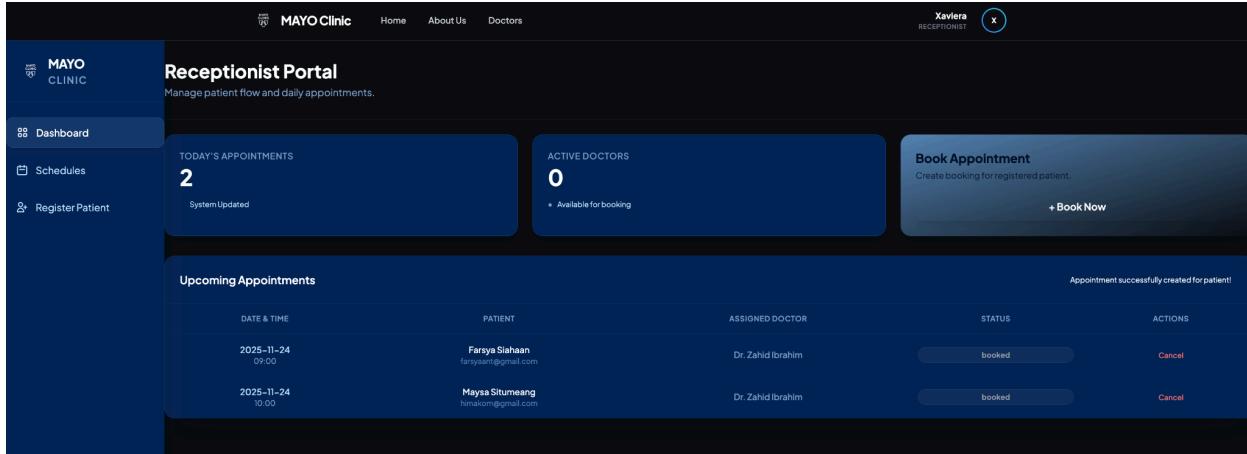
Have questions about our services or need emergency assistance? We are here to help 24/7.

 Emergency Line
(0274) 123 - 4567

 Email Support
help@mayoclinic.com

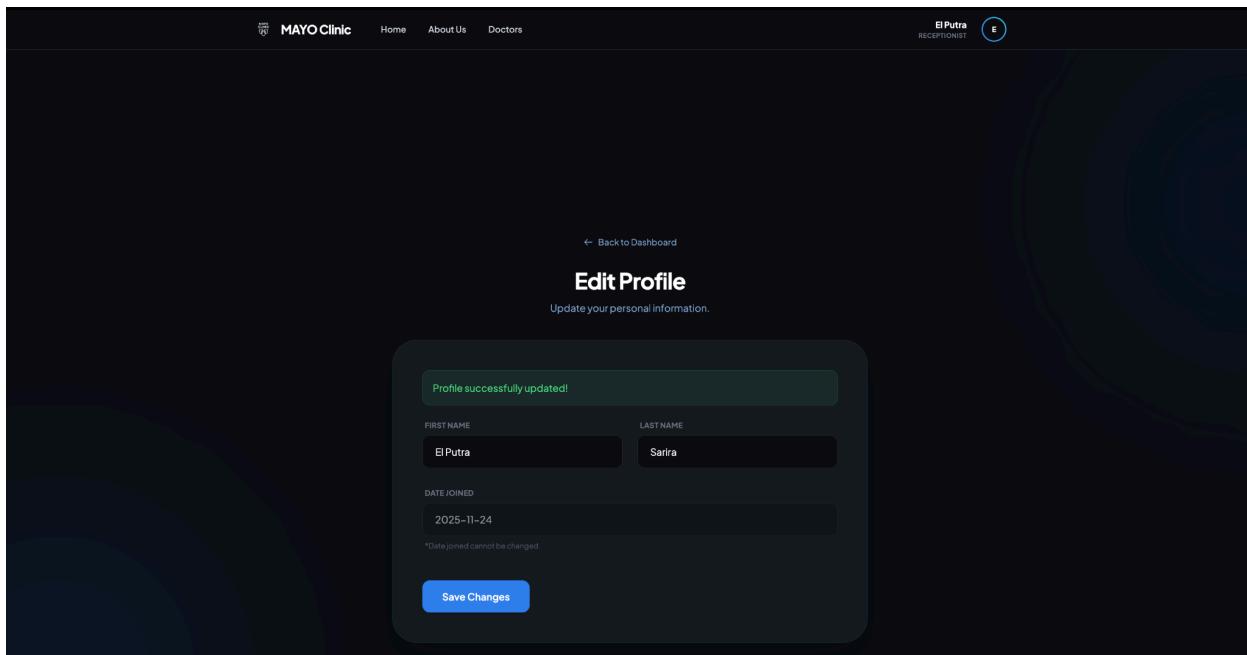
Resepsonis

Tampilan *dashboard* resepsionis menyediakan akses menu melalui *sidebar*, yaitu: *Dashboard*, *Schedules* (jadwal dokter), dan *Register Patient* (pendaftaran pasien).



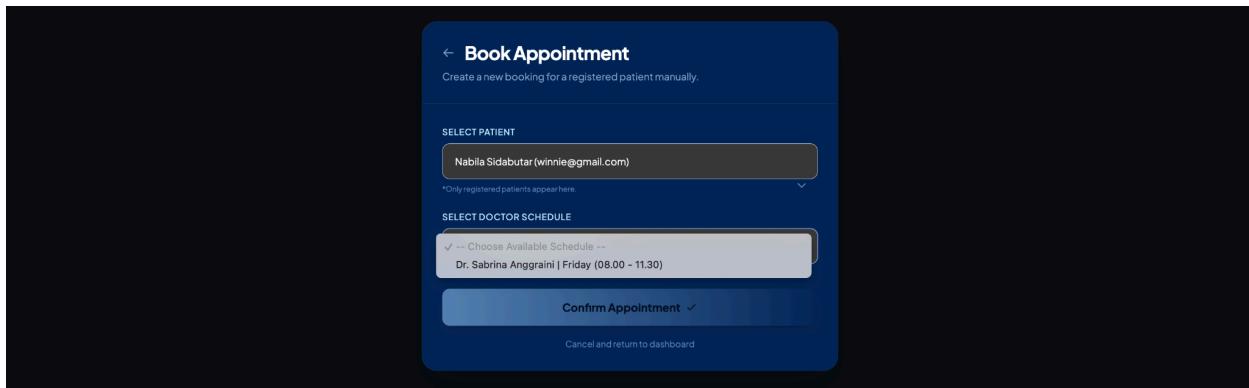
Gambar 2. *Dashboard* Resepsonis

Resepsonis dapat memperbarui nama akun melalui fitur *Edit Profile*. Perubahan data akan otomatis tersinkronisasi pada tampilan utama *dashboard*.



Gambar 3. *Edit Profil* Resepsonis (Xaviera → El Putra)

Fitur *Book Appointment* digunakan untuk mendaftarkan janji temu baru dengan memilih pasien terdaftar dan jadwal dokter yang tersedia.



Gambar 4. Book Appointment Melalui Resepsionis

Data appointment yang berhasil ditambahkan akan langsung muncul pada tabel *Upcoming Appointments* di dashboard.

Upcoming Appointments				
DATE & TIME	PATIENT	ASSIGNED DOCTOR	STATUS	ACTIONS
2025-11-24 08:00	Nabila Sidabutar winnie@gmail.com	Dr. Sabrina Anggraini	booked	Cancel
2025-11-24 09:00	Farsya Slaahan farsya@gmail.com	Dr. Zahid Ibrahim	booked	Cancel
2025-11-24 10:00	Maysa Situmeang limakon@gmail.com	Dr. Zahid Ibrahim	booked	Cancel

Gambar 5. Appointment Berhasil Ditambahkan

Resepsionis dapat membatalkan jadwal menggunakan tombol *Cancel*, yang secara otomatis menghapus data tersebut dari sistem. Sementara angka pada *Active Doctors* merepresentasikan jumlah dokter yang jadwalnya masih tersedia pada hari tersebut.

Upcoming Appointments				
DATE & TIME	PATIENT	ASSIGNED DOCTOR	STATUS	ACTIONS
2025-11-24 08:00	Nabila Sidabutar winnie@gmail.com	Dr. Sabrina Anggraini	booked	Cancel

Gambar 6. Appointment Berhasil Dibatalkan

