BAB I
PENDAHULUAN

1.1. Latar Belakang

Penyakit Kardiovaskular (*Cardiovascular Disease* atau CVD) sampai saat ini masih menjadi salah satu penyebab kematian tertinggi di dunia. Organisasi Kesehatan Dunia (WHO) melaporkan bahwa penyakit kardiovaskular bertanggung jawab atas kematian belasan juta jiwa setiap tahunnya [1]. Secara medis, penyakit ini muncul karena penumpukan kondisi buruk pada tubuh seperti merokok, darah tinggi, kolesterol, dan riwayat diabetes [2]. Di Indonesia, penyakit ini menempati urutan pertama sebagai penyebab kematian tertinggi dengan menelan lebih dari 250.000 jiwa per tahun dan menguras anggaran BPJS Kesehatan hingga belasan triliun rupiah [3]. Kondisi ini menunjukkan bahwa masyarakat butuh alat deteksi dini yang praktis, sejalan dengan target *Sustainable Development Goals* (SDGs) poin ketiga tentang kesehatan dan kesejahteraan [4].

Untuk merespons masalah tersebut, dibuatlah sebuah aplikasi kesehatan bernama "Selaras" yang menggunakan algoritma medis SCORE2 dan turunannya untuk menghitung risiko penyakit serangan jantung pada populasi Asia-Pasifik [5]. Namun, hasil kalkulasi algoritma medis yang sekadar menampilkan angka persentase risiko sering kali memicu kebingungan dan sulit dipahami oleh masyarakat awam [6]. Oleh karena itu, aplikasi Selaras mengintegrasikan *Generative AI* melalui Gemini API sebagai penerjemah virtual untuk mengubah angka kaku tersebut menjadi *health coaching* yang personal dan mudah dipahami [7].

Pengembangan fitur cerdas tersebut ternyata menghadirkan tantangan teknis yang cukup berat pada arsitektur *Back-End*, khususnya dalam menangani beban komputasi ganda antara kalkulasi matematis dan integrasi API eksternal. Apabila dipaksakan menggunakan arsitektur *Model-View-Controller* (MVC) bawaan pada Laravel 12, kompleksitas algoritma medis SCORE2 dan turunannya dapat menimbulkan penumpukan baris kode pada lapisan *Controller* (*Fat Controller*). Kondisi ini secara langsung melanggar kaidah *Clean Architecture*, membuat basis

kode menjadi sangat kaku, sulit dipelihara (*unmaintainable*), dan mustahil untuk diuji fungsionalitasnya secara terisolasi (*untestable*) [8]. Oleh karena itu, penerapan *Service-Repository Pattern* menjadi solusi arsitektural guna memisahkan dan mengisolasi logika kalkulasi medis dari *Controller*. Pemisahan ini memastikan sistem mematuhi *Single Responsibility Principle (*SRP), sehingga algoritma medis dapat divalidasi dan dikembangkan secara independen tanpa mengganggu alur komunikasi data [9].

Tantangan arsitektur *Back-End* selanjutnya muncul dari sisi integrasi Gemini API yang berpotensi memicu lonjakan biaya *token* kecerdasan buatan [10], serta ancaman pelanggaran privasi data medis atau *Personally Identifiable Information* (PII) jika dikirimkan secara mentah [11]. Untuk mengatasi batasan tersebut, sistem menerapkan teknik *Context Injection* dan *sliding window* menggunakan data dari MySQL guna menyensor data sensitif menjadi anonim dan membatasi riwayat yang dikirim ke AI [12]. Sebagai penyempurnaan di lingkungan *shared hosting*, pembaruan data program kesehatan harian dijalankan secara *asynchronous* menggunakan otomasi *Cron Job* agar tidak membebani performa antarmuka pengguna. Berdasarkan permasalahan teknis *Back-End* tersebut, penelitian Tugas Akhir ini mengambil judul "Implementasi Backend Aplikasi Selaras: Integrasi Algoritma SCORE2 dan Gemini API untuk Estimasi Risiko Jantung".

1.2 Perumusan masalah

Berdasarkan latar belakang yang telah diuraikan, rumusan masalah dalam penelitian ini difokuskan pada implementasi dan optimasi arsitektur *Back-End* aplikasi Selaras. Adapun rincian masalah teknis yang akan diselesaikan adalah:

1. Sejauh mana penerapan *Service-Repository Pattern* pada Laravel 12 mampu mengisolasi kompleksitas algoritma SCORE2 dan turunannya guna menghasilkan API yang tervalidasi fungsionalitasnya tanpa membebani lapisan *Controller*?
2. Apa dampak pemanfaatan *Context Injection* dan *sliding window* pada integrasi Gemini API terhadap efisiensi penggunaan *token* serta keberhasilan penyensoran privasi data medis (PII) pengguna?

3. Seperti apa model rancangan fungsionalitas otomasi *Cron Job* pada lingkungan *shared hosting* untuk mengelola siklus program "Langkah Sehat" 28 hari secara *asynchronous*?

1.3 Tujuan Penelitian

Sejalan dengan rumusan masalah di atas, penelitian ini bertujuan untuk menyelesaikan tantangan teknis pada sisi *server*, dengan rincian sebagai berikut:

1. Mengevaluasi tingkat keberhasilan penerapan *Service-Repository Pattern* pada Laravel 12 guna mengisolasi kompleksitas algoritma SCORE2 dan turunannya, sehingga menghasilkan API yang tervalidasi fungsionalitasnya tanpa membebani lapisan *Controller*.
2. Menganalisis dampak pemanfaatan *Context Injection* dan *sliding window* pada integrasi Gemini API terhadap efisiensi penggunaan *token* serta keberhasilan penyensoran privasi data medis (PII) pengguna.
3. Merumuskan model rancangan fungsionalitas otomasi *Cron Job* pada lingkungan *shared hosting* untuk mengelola siklus program "Langkah Sehat" 28 hari secara *asynchronous*.

1.4 Batasan Masalah

Agar penelitian ini lebih terarah dan relevan dengan tujuan yang ingin dicapai, ruang lingkup pembahasan dibatasi pada poin-poin berikut:

1. Fokus pengembangan secara murni pada pembuatan sistem *Back-End* menggunakan Laravel 12, tanpa membahas riset UI/UX maupun penulisan kode *Front-End*.
2. Arsitektur perangkat lunak dibatasi pada penerapan *Service-Repository Pattern* untuk mengelola *query database* MySQL dan pemisahan logika dari *Controller*.
3. Parameter estimasi risiko kardiovaskular secara spesifik menggunakan algoritma klinis SCORE2 dan turunannya dengan standar perhitungan resmi untuk wilayah Asia-Pasifik.
4. Integrasi *Generative AI* hanya menggunakan Gemini API via *Context Injection* dan *sliding window* di MySQL, tanpa menggunakan infrastruktur tambahan seperti *Vector Database* atau *Retrieval-Augmented Generation*.

5. Keseluruhan implementasi sistem, termasuk *Cron Job*, disesuaikan dengan batas spesifikasi server *shared hosting*.
6. Implementasi keandalan dan keamanan *Application Programming Interface* (API) pada *server* mencakup penerapan standar ketahanan sistem dasar, meliputi autentikasi menggunakan Laravel Sanctum, *Rate Limiting*, serta mekanisme *fallback* dan pembersihan format data (*Regex parsing*) untuk memitigasi anomali pada layanan pihak ketiga.
7. Validasi akurasi perhitungan SCORE2 dilakukan dengan komparasi output terhadap matriks tabel risiko standar dari *European Society of Cardiology* (ESC) menggunakan dataset klinis sintetis.

1.5 Manfaat Penelitian

Implementasi arsitektur *Back-End* pada aplikasi Selaras ini diharapkan dapat memberikan manfaat praktis maupun akademis, di antaranya:

1. Bagi Pengguna Akhir (Masyarakat)

   Memberikan akses platform deteksi dini kardiovaskular yang komunikatif. Pengguna mendapatkan hasil persentase risiko yang diterjemahkan menjadi narasi *health coaching* oleh AI, dengan jaminan keamanan data medis yang ketat.
2. Bagi Fasilitas Kesehatan

   Menyediakan alat deteksi dini mandiri yang andal untuk membantu menekan beban antrean di Faskes Tingkat I, sejalan dengan target pencapaian *Sustainable Development Goals* (SDGs) poin ketiga.
3. Bagi Pengembangan Ilmu Pengetahuan

   Memberikan bukti nyata terkait keandalan *Service-Repository Pattern* pada Laravel 12 di *shared hosting*, serta mendemonstrasikan efisiensi teknik *Context Injection* dan *sliding window* pada MySQL dalam mengoptimalkan biaya *token* AI berskala menengah.
4. Bagi Validasi Praktis Rekayasa Perangkat Lunak

   Menyediakan *prototype* arsitektur sistem berstandar kompetisi nasional telah divalidasi secara praktis dan akademis, dibuktikan melalui pencapaian Juara 3 pada kompetisi *App Development* di ajang Multimedia and Game Event (MAGE) 11, Institut Teknologi Sepuluh Nopember (ITS).