

RANCANG BANGUN APLIKASI WEB UNTUK MENGUKUR TINGKAT STRES MAHASISWA

Doni Wicaksono
Rekayasa Perangkat Lunak
Universitas Telkom Purwokerto
Purwokerto, Indonesia
doniwicaksono@student.telkomuniversity.ac.id

Arif Amrulloh, S.Kom., M.Kom
Rekayasa Perangkat Lunak
Universitas Telkom Purwokerto
Purwokerto, Indonesia
amrulloh@telkomuniversity.ac.id

Maryona Septiara, S.Pd., M.Kom
Rekayasa Perangkat Lunak
Universitas Telkom Purwokerto
Purwokerto, Indonesia
septiara@telkomuniversity.ac.id

Abstrak — Stres pada mahasiswa merupakan salah satu persoalan yang umum terjadi dan dapat memengaruhi prestasi akademik maupun kesehatan mental jika tidak ditangani sejak dini. Penelitian ini bertujuan untuk merancang dan membangun sistem berbasis web yang dapat membantu mahasiswa dalam mengukur tingkat stres secara mandiri melalui algoritma *Certainty Factor*. Metode pengembangan yang digunakan adalah *Waterfall Model*, tahapan pengembangan meliputi pengumpulan data melalui kuesioner, wawancara pakar untuk menyusun basis pengetahuan, perancangan sistem berbasis *use case* dan *activity diagram*, pembangunan antarmuka menggunakan React dan Next.js. Sistem memiliki fitur asesmen tingkat stres, pengelolaan hasil asesmen, autentikasi, serta *dashboard* pengguna dan admin. Pengujian dilakukan dengan metode uji fungsional dan *usability*, dan menunjukkan bahwa seluruh fitur utama berjalan sesuai skenario serta memperoleh skor rata-rata *usability* sebesar 81,94. Sistem ini diharapkan dapat menjadi alat bantu yang bermanfaat bagi mahasiswa dalam mengenali potensi stres yang dialami secara praktis dan terstruktur.

Kata kunci — tingkat stres, *certainty factor*, *waterfall model*, *black box testing*, *system usability scale*

I. PENDAHULUAN

Mahasiswa sebagai aset berharga di masa depan berperan penting bagi kemajuan suatu bangsa. Tidak hanya menuntut ilmu di bangku pendidikan, mereka juga diharapkan untuk memberikan kontribusi nyata dalam memerangi tantangan yang dihadapi oleh masyarakat dan negara. Namun demikian, mahasiswa juga memiliki tantangan tersendiri dalam memerangi masalah di lingkungan akademik. Mereka merupakan kelompok populasi yang rentan mengalami stres karena tuntutan akademik seperti ujian, jadwal yang ketat dan konflik interpersonal [1]. Masalah stres dan gangguan kesehatan mental dapat dialami oleh siapapun, tak terkecuali mahasiswa di Universitas Telkom Purwokerto.

Berdasarkan hasil dari kuesioner yang dilakukan pada sejumlah mahasiswa di Universitas Telkom Purwokerto, didapatkan 60% mahasiswa merasa terbebani oleh tugas,

56,7% mahasiswa merasa sulit berkonsentrasi ketika belajar, dan 66,7% mahasiswa merasa cemas mengenai nilai akademik. Stres yang tidak terkelola dapat berdampak negatif pada kesehatan fisik dan mental mahasiswa, serta mengganggu kinerja akademik mereka [2]. Beban tugas yang berlebihan membebani mahasiswa dengan tuntutan yang melebihi kapasitas, tekanan sosial dari keluarga dapat membuat mahasiswa merasa tidak mampu memenuhi harapan [3]. Permasalahan-permasalahan tersebut berpotensi menjadi gangguan untuk kondisi kesehatan mental mahasiswa.

Aplikasi ini mengadopsi algoritma *Certainty Factor* dan metode *Waterfall* untuk kebutuhan perancangan dan pembangunannya. Algoritma *Certainty Factor* digunakan untuk menyelesaikan ketidakpastian dengan menghitung nilai kemungkinan dari semua faktor yang terlibat dalam suatu masalah [4]. Di sisi lain, menggunakan *Waterfall* dalam proyek pengembangan perangkat lunak dapat memperkirakan waktu penyelesaian proyek dan fase dengan akurat, serta mengidentifikasi masalah efisiensi yang berpotensi akibat tingkat sumber daya yang tidak optimal [5]. *Black Box testing* digunakan untuk pengujian fungsional yang di mana dapat mendeteksi pelanggaran spesifikasi sistem dan menghasilkan rangkaian pengujian dengan cakupan sistem yang baik dan kapabilitas dalam *error-detection* [6]. Sementara pengujian *usability* menggunakan *System Usability Scale* yang merupakan alat yang mudah digunakan untuk menilai kegunaan produk, dengan skor dari 0 (negatif) hingga 100 (positif) [7].

II. KAJIAN TEORI

Bab ini menguraikan landasan teoretis yang menjadi dasar dalam perancangan dan pengembangan aplikasi. Pembahasan mencakup konsep-konsep kunci terkait teknologi yang digunakan, metodologi pengembangan perangkat lunak, serta pendekatan untuk pengukuran dan evaluasi sistem. Teori-teori ini berfungsi sebagai kerangka acuan untuk memastikan bahwa setiap tahapan penelitian, mulai dari pembangunan hingga pengujian, didasarkan pada praktik dan prinsip yang relevan dan teruji.

A. Rancang Bangun

Rancang bangun adalah proses kreatif di mana persyaratan stakeholder, kebutuhan bisnis, dan pertimbangan teknis bersatu untuk merumuskan produk atau sistem. Proses ini melibatkan pembuatan representasi atau model perangkat lunak yang merinci arsitektur, struktur data, antarmuka, dan komponennya, yang sangat penting untuk mengimplementasikan sistem [8].

B. Aplikasi Web

Aplikasi web adalah sistem informasi yang dapat memberikan konten dan fungsionalitas yang kompleks kepada populasi pengguna akhir yang luas dan terdiri dari sekumpulan halaman web yang dibuat sebagai respons terhadap permintaan pengguna [7].

C. Waterfall Model

Waterfall merupakan model pertama dari proses pengembangan perangkat lunak yang dipublikasikan berasal dari model proses rekayasa yang digunakan dalam sistem rekayasa militer besar. Karena adanya aliran berurutan dari satu fase ke fase berikutnya, model ini dikenal sebagai model air terjun atau *software life cycle*. *Waterfall* merupakan contoh proses yang didorong oleh perencanaan. Secara prinsip, model ini merencanakan dan menjadwalkan semua aktivitas proses sebelum memulai pengembangan perangkat lunak [9].

D. Certainty Factor

Certainty Factor atau CF merupakan nilai untuk mengukur keyakinan pakar. CF menunjukkan ukuran kepastian terhadap suatu fakta atau aturan. Nilai tertinggi dalam CF adalah +1,0 (pasti benar atau *definitely*), dan nilai terendah dalam CF adalah -1,0 (pasti salah atau *definitely not*). Nilai positif merepresentasikan derajat keyakinan, sedangkan nilai negatif merepresentasikan derajat ketidakpercayaan. *Certainty Factor* didefinisikan sebagai berikut:

$$CF[h, e] = MB[h, e] - MD[h, e] \quad (1)$$

$MB[h, e]$ adalah ukuran tingkat keyakinan terhadap hipotesis h , jika dipengaruhi eviden e (antara 0 dan 1), $MD[h, e]$ adalah ukuran tingkat ketidakpercayaan terhadap hipotesis h , jika dipengaruhi eviden e (antara 0 dan 1). Sebagai contoh jika seorang pakar menyatakan beberapa eviden (fakta) adalah hampir pasti benar (*almost certainty*), maka dapat diberikan nilai CF 0,8 pada eviden ini [10].

E. Wireframe

Wireframe mengacu pada sketsa desain antarmuka pengguna yang membantu membangun pemahaman yang realistis tentang ruang desain dan mendapatkan inspirasi desain dari aplikasi yang ada [11].

F. React

React adalah sebuah *framework* untuk membangun aplikasi web yang bertujuan untuk mengatasi masalah seperti kode yang rapuh, desain non-modular, dan pembaruan antarmuka tambahan sebagai respons terhadap perubahan state [12].

G. Firebase

Firebase adalah solusi *Platform-as-a-Service* (PaaS) terkemuka dalam industri pengembangan aplikasi, yang digunakan oleh jutaan aplikasi seluler dan situs web [13].

H. Unified Modeling Language

Unified Modeling Language (UML) adalah bahasa standar untuk menulis cetak biru perangkat lunak. UML dapat digunakan untuk memvisualisasikan, menentukan, membangun, dan mendokumentasikan artefak dari sistem perangkat lunak yang intensif [14].

I. System Usability Scale

System Usability Scale (SUS) adalah alat yang dikembangkan oleh John Brooke pada tahun 1986 di Digital Equipment Corporation untuk menilai kegunaan yang dirasakan dari suatu sistem atau produk. Alat ini terdiri dari 10 pernyataan, dengan pengguna yang menilai tingkat persetujuan mereka pada skala lima poin. Pernyataan-pernyataan tersebut merupakan campuran dari kata-kata positif dan negatif [15].

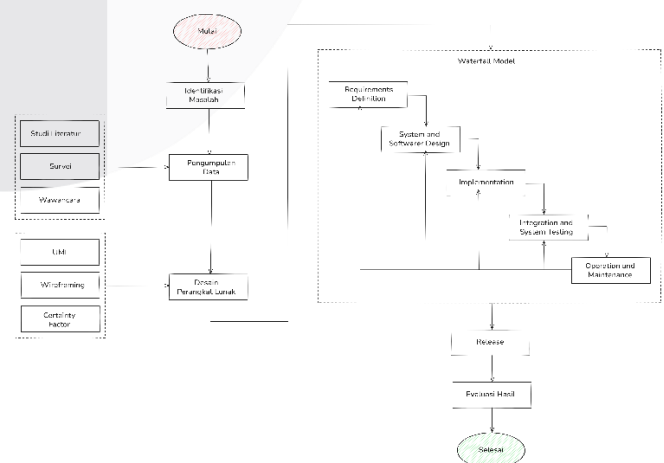
J. Black Box Testing

Black Box testing, juga disebut *behavioral testing*, berfokus pada persyaratan fungsional perangkat lunak. *Black Box testing* memungkinkan *software engineer* untuk mendapatkan serangkaian kondisi input yang akan sepenuhnya menjalankan semua persyaratan fungsional untuk sebuah program [16].

III. METODE

Proses penelitian ini berlangsung selama sembilan bulan, dimulai dari studi literatur hingga evaluasi hasil, dengan fokus pada prosedur identifikasi masalah, pengumpulan data, desain perangkat lunak, dan implementasi sistem menggunakan *Waterfall Model*.

Untuk memberikan gambaran visual yang jelas mengenai keseluruhan proses penelitian, berikut disajikan diagram alir penelitian yang menggambarkan tahapan-tahapan kunci mulai dari identifikasi masalah hingga evaluasi sistem.



GAMBAR 1
(Diagram Alir Penelitian)

A. Identifikasi Masalah

Pada tahap ini, dilakukan analisis mendalam terhadap isu-isu relevan untuk merumuskan masalah utama yang akan diselesaikan. Proses ini mencakup observasi awal dan studi pendahuluan untuk memastikan urgensi dan relevansi masalah penelitian, sehingga diperoleh rumusan masalah yang spesifik dan terukur sebagai acuan untuk tahapan selanjutnya.

B. Pengumpulan Data

Tahapan ini bertujuan untuk menghimpun informasi komprehensif yang diperlukan dalam perancangan solusi, dilaksanakan melalui triangulasi data yang terdiri dari studi literatur untuk membangun landasan teoretis, survei untuk memperoleh data kuantitatif mengenai persepsi dan kebutuhan pengguna, serta wawancara dengan pakar psikolog klinis untuk mendapatkan pemahaman kualitatif yang mendalam dan masukan ahli.

C. Desain Perangkat Lunak

Data yang telah terkumpul dianalisis dan ditransformasikan menjadi sebuah rancangan perangkat lunak yang fungsional dan sistematis. Proses desain ini menerjemahkan kebutuhan pengguna menjadi spesifikasi teknis yang siap untuk diimplementasikan melalui penggunaan *Unified Modeling Language* (UML) untuk memodelkan arsitektur dan alur kerja sistem, *wireframing* untuk mengembangkan rancangan tampilan aplikasi sebagai kerangka visual *low-fidelity*, serta implementasi algoritma *Certainty Factor* untuk merancang logika inferensi sistem sebagai basis pengetahuan.

D. Requirements Definition

Pada fase ini, seluruh kebutuhan fungsional dan non-fungsional perangkat lunak didefinisikan secara rinci dan terdokumentasi, berfungsi sebagai acuan utama bagi tim pengembang untuk seluruh proses selanjutnya.

E. System and Software Design

Fase ini melibatkan perancangan teknis arsitektur sistem, struktur data, dan desain modular perangkat lunak berdasarkan kebutuhan yang telah didefinisikan, membentuk fondasi struktural bagi implementasi.

F. Implementation

Spesifikasi desain diterjemahkan ke dalam kode program yang dapat dieksekusi, di mana setiap modul perangkat lunak dikembangkan secara terpisah untuk membangun solusi perangkat lunak yang fungsional.

G. Integration and System Testing

Setelah implementasi, unit program individu atau program-program tersebut diintegrasikan dan diuji sebagai sistem yang utuh untuk memastikan bahwa persyaratan perangkat lunak telah terpenuhi, serta mengidentifikasi dan memperbaiki kecacatan sebelum diserahkan kepada pelanggan.

H. Operation and Maintenance

Ini adalah fase siklus hidup yang paling lama, di mana sistem dipasang dan digunakan secara praktis, dengan pemeliharaan yang meliputi perbaikan kesalahan yang tidak terdeteksi pada tahap-tahap awal, peningkatan implementasi

unit-unit sistem, dan peningkatan layanan seiring dengan penemuan persyaratan baru.

I. Release

Setelah perangkat lunak dinyatakan stabil dan lulus dari fase pengujian sistem, tahap ini melibatkan distribusi dan implementasi perangkat lunak yang telah dikembangkan pada lingkungan operasional yang sesungguhnya untuk digunakan oleh pengguna akhir.

J. Evaluasi Hasil

Tahap final dari alur penelitian ini bertujuan untuk mengukur tingkat keberhasilan perangkat lunak, mencakup pengujian *usability* perangkat lunak yang diukur secara kuantitatif menggunakan *System Usability Scale* (SUS) untuk menilai tingkat kemudahan penggunaan dan kepuasan pengguna, dengan hasil gabungan menjadi dasar untuk penarikan kesimpulan.

IV. HASIL DAN PEMBAHASAN

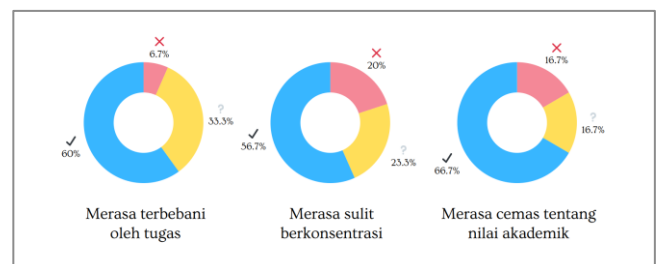
Bab ini menyajikan secara komprehensif temuan-temuan kunci yang diperoleh dari seluruh tahapan penelitian, mulai dari pengumpulan data awal hingga implementasi dan evaluasi sistem. Bagian ini akan menguraikan hasil analisis kebutuhan pengguna, detail perancangan dan pengembangan aplikasi web pengukuran tingkat stres menggunakan *Waterfall Model* dan algoritma *Certainty Factor*, serta menyajikan dan mendiskusikan hasil pengujian fungsionalitas (*Black Box Testing*) dan *usability* (*System Usability Scale*). Pembahasan akan fokus pada bagaimana sistem yang dibangun memenuhi tujuan penelitian dan implikasinya terhadap pemahaman tingkat stres mahasiswa.

A. Identifikasi Masalah

Fenomena stres yang tidak terkelola dengan baik di kalangan mahasiswa, yang memiliki dampak serius dan merugikan pada kesehatan mental serta prestasi akademik mereka. Stres yang berkepanjangan dapat memicu berbagai masalah psikologis seperti kecemasan, depresi, dan gangguan tidur, yang pada gilirannya menurunkan kemampuan konsentrasi, motivasi belajar, dan akhirnya menghambat pencapaian akademik mahasiswa secara optimal.

B. Pengumpulan Data

Untuk mengumpulkan data, sebuah kuesioner disebarikan kepada 30 mahasiswa aktif dari berbagai program studi di Universitas Telkom Purwokerto, termasuk S1 Perangkat Lunak, S1 Sistem Informasi, dan S1 Teknik Informatika. Hasil rangkuman dari pernyataan-pernyataan kunci disajikan sebagai berikut:



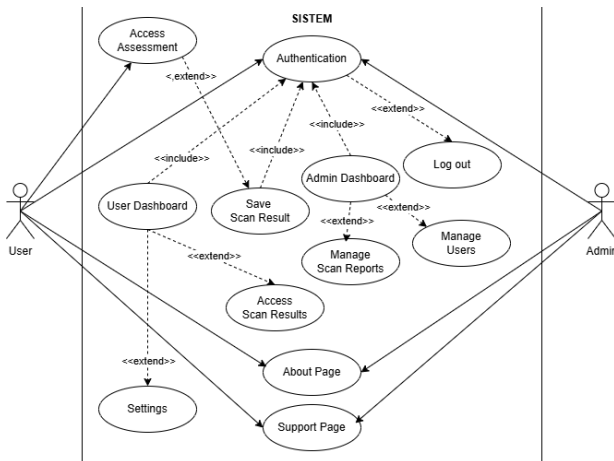
GAMBAR 2
(Hasil Kuesioner Penelitian)

C. Desain Perangkat Lunak

Pada tahap ini, perilaku sistem dimodelkan menggunakan *Unified Modeling Language* (UML), dengan *use case diagram* untuk mendefinisikan interaksi aktor dan *activity diagram* untuk memetakan alur kerja. Selain itu, *wireframe* digunakan sebagai representasi visual untuk memandu pengembangan antarmuka pengguna. Proses perancangan juga mencakup pembuatan basis pengetahuan untuk logika sistem dengan algoritma *Certainty Factor*.

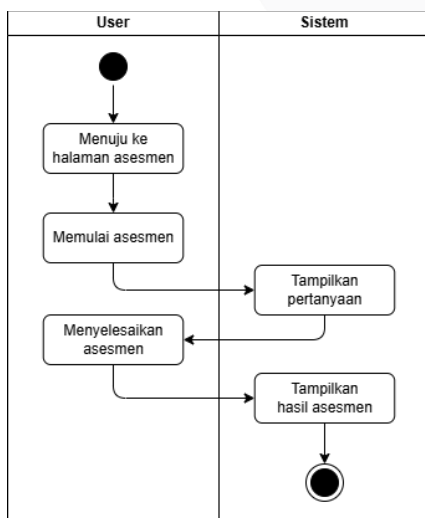
a. Use Case Diagram

Sistem ini dirancang dengan dua jenis pengguna utama: pengguna (*user*) dan admin, masing-masing dengan tingkat akses fitur yang berbeda. Pengguna dapat langsung melakukan asesmen tanpa perlu otentikasi, namun fitur seperti penyimpanan hasil, *dashboard*, pengaturan akun, dan riwayat asesmen memerlukan *log in*. Setelah *log in*, pengguna dapat mengakses fitur-fitur tersebut dan memiliki opsi untuk *log out*. Baik pengguna maupun admin juga dapat mengakses halaman informasi seperti *about* dan *support* tanpa perlu *log in*. Berbeda dengan pengguna, admin memiliki fokus pada pengelolaan sistem, memerlukan *log in* untuk mengakses *dashboard* admin, melihat laporan asesmen, dan mengelola data pengguna.



GAMBAR 3
(Use Case Diagram)

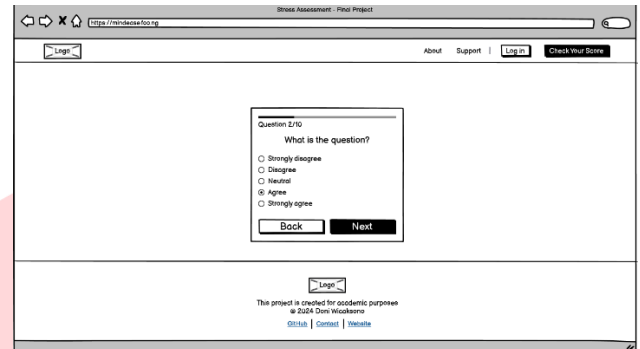
b. Activity Diagram



GAMBAR 4
(Activity Diagram untuk Asesmen)

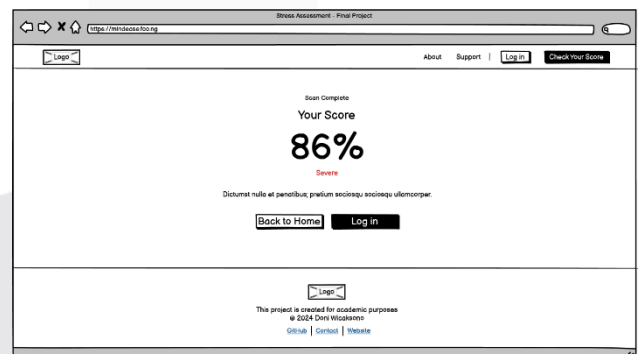
Proses asesmen dimulai saat pengguna masuk ke halaman asesmen dan memulai serangkaian pertanyaan berdasarkan gejala yang telah dikelompokkan sebelumnya. Sistem akan menampilkan pertanyaan secara berurutan hingga asesmen selesai. Setelah itu, hasil asesmen akan ditampilkan berdasarkan perhitungan menggunakan algoritma *Certainty Factor*.

c. Wireframe



GAMBAR 5
(Wireframe halaman Kuis Asesmen)

Halaman kuis pengukuran tingkat stres ini didesain untuk mengumpulkan data pengguna melalui serangkaian pertanyaan. Di tengah layar, terdapat kotak dialog interaktif berisi pernyataan dengan opsi jawaban berbentuk skala Likert. Bagian atas kotak dialog menampilkan indikator progres yang menunjukkan posisi pengguna dalam total pertanyaan. Di bawah indikator tersebut, pertanyaan utama disajikan bersama lima opsi jawaban yang tersusun vertikal. Tombol *back* dan *next* ditempatkan di bagian bawah kotak dialog untuk navigasi antar pertanyaan.



GAMBAR 6
(Wireframe halaman Hasil Asesmen)

Setelah kuis selesai, halaman berikutnya menampilkan hasil akhir pengukuran tingkat stres. Skor penilaian disajikan secara mencolok di tengah halaman, dilengkapi dengan label kategori yang menginterpretasikan nilai tersebut. Di bawah skor, terdapat deskripsi singkat yang menawarkan saran berdasarkan hasil penilaian. Untuk memudahkan navigasi pengguna, dua tombol aksi tersedia: tombol *back to home* untuk kembali ke halaman utama, dan tombol *log in* untuk mengakses fitur tambahan yang memerlukan autentikasi.

d. Certainty Factor

Berdasarkan wawancara dengan pakar psikolog klinis, gejala-gejala stres dikelompokkan menjadi empat kategori

utama untuk asesmen tingkat stres: fisik, kognitif, emosional, dan perilaku. Gejala fisik mencakup ketegangan otot dan tubuh, pola tidur terganggu, serta berkurangnya energi yang menyebabkan mudah lelah. Gejala kognitif meliputi penurunan konsentrasi dan perhatian, serta kesulitan membuat keputusan. Gejala emosional ditandai dengan ketidakmampuan untuk santai, mudah marah, dan perasaan khawatir. Gejala perilaku mengidentifikasi perubahan pola perilaku akibat stres, seperti kewalahan dengan tugas dan tekanan tenggat waktu.

Setiap kategori gejala diberikan bobot yang berbeda berdasarkan tingkat signifikansinya dalam diagnosis stres. Penentuan bobot ini didasarkan pada pakar dan relevansi klinis masing-masing kategori gejala.

TABEL 1
(Kategori Gejala dan Bobot CF)

Kategori	Kode Gejala	Deskripsi	Bobot CF
Fisik	q3	Ketegangan otot dan tubuh	0.8
	q4	Pola tidur terganggu	1.0
	q8	Berkurangnya energi menjadi mudah lelah	0.8
Kognitif	q5	Konsentrasi dan perhatian berkurang	1.0
	q9	Kesulitan membuat keputusan	0.8
Emosional	q2	Tidak dapat santai	0.8
	q6	Mudah marah	0.8
	q7	Merasa khawatir	0.6
Perilaku	q1	Kewalahan dengan tugas	0.6
	q10	Tekanan tenggat waktu	0.8

D. Requirements Definition

Semua kebutuhan sistem ditetapkan secara menyeluruh sebagai dasar untuk fase perancangan dan implementasi. Kebutuhan ini diperoleh dari analisis data kuesioner dan karakteristik mahasiswa aktif Universitas Telkom Purwokerto sebagai target pengguna. Spesifikasi kebutuhan sistem kemudian diklasifikasikan menjadi dua kategori utama:

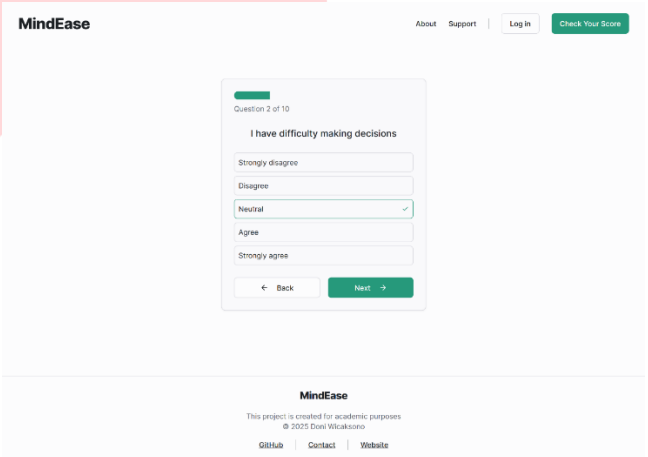
TABEL 2
(Kebutuhan Sistem)

Kategori	Deskripsi
Kebutuhan Fungsional	Sistem menyediakan asesmen tingkat stres yang mudah diakses pengguna.
	Sistem memproses data input menggunakan algoritma CF untuk diagnosis tingkat stres.
	Sistem menampilkan hasil diagnosis segera setelah asesmen selesai.
	Sistem dapat menyimpan riwayat hasil asesmen untuk melacak perkembangan tingkat stres pengguna.
Kebutuhan Non-Fungsional	Antarmuka sistem harus sederhana, intuitif, dan user-friendly.
	Sistem harus responsif dan dapat diakses dari berbagai perangkat (desktop, laptop, smartphone).
	Pengguna dapat langsung menggunakan fitur asesmen tanpa registrasi atau log in.

E. System and Software Design

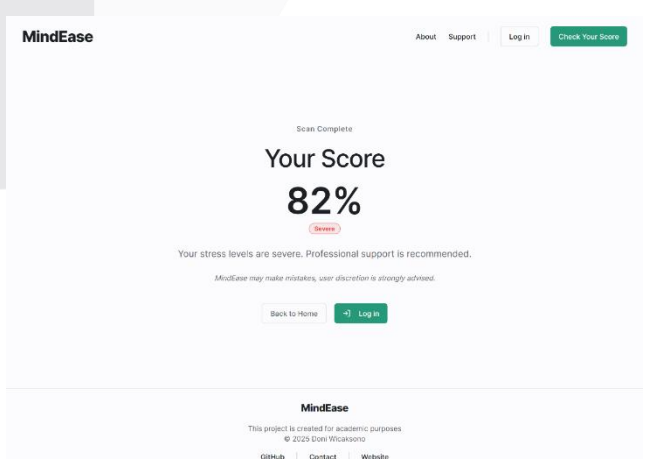
Perancangan sistem dan perangkat lunak ini mengadopsi arsitektur web modern yang memisahkan logika antarmuka pengguna (client-side) dari pengelolaan data (backend). Untuk client-side, antarmuka pengguna dirancang menggunakan framework Next.js 15 yang berbasis pada library React 19, dipilih karena kemampuannya dalam menciptakan struktur aplikasi modular, mengelola state, dan navigasi halaman secara efisien untuk aplikasi interaktif.

Sementara itu, untuk arsitektur backend dan persistensi data, sistem ini mengandalkan Google Firebase; Firebase Authentication menangani autentikasi pengguna secara aman, dan Cloud Firestore berfungsi sebagai basis data utama yang fleksibel dan terstruktur untuk menyimpan hasil asesmen, riwayat, serta informasi akun pengguna. Implementasi dari arsitektur dan desain perangkat lunak ini terwujud dalam antarmuka pengguna yang fungsional. Hasil dari tahap wireframing diuraikan sebagai berikut:



GAMBAR 7
(Tampilan halaman Kuis Asesmen)

Kuis asesmen diawali dengan pengguna diarahkan ke halaman yang menampilkan serangkaian pertanyaan berdasarkan gejala yang telah dikelompokkan sebelumnya. Di bagian atas, indikator progres akan menunjukkan posisi pengguna (misalnya, pertanyaan kedua dari sepuluh).



GAMBAR 8
(Tampilan halaman Hasil Asesmen)

Setelah asesmen selesai, hasilnya ditampilkan dengan skor persentase besar di tengah layar, disertai label kategori yang mengindikasikan tingkat stres pengguna. Di bawah skor tersebut, tersedia saran untuk mencari dukungan profesional jika diperlukan.

F. Implementation

Implementasi proyek ini mengadopsi arsitektur berbasis komponen React 19 yang dipadukan dengan *framework* Next.js 15 untuk menghasilkan aplikasi web yang *scalable* dan *maintainable*. Setiap modul dikembangkan secara terpisah, mengikuti prinsip *separation of concerns*, di mana logika bisnis, presentasi, dan manajemen *state* dipisahkan ke dalam lapisan-lapisan yang berbeda.

```
import React, { ReactNode } from 'react';
import { Inter } from 'next/font/google';

const inter = Inter({ subsets: ['latin'] });

interface LayoutProps {
  children: ReactNode;
  title?: string;
}

export default function Layout({ children, title = 'MindEase App' }: LayoutProps) {
  return (
    <html lang="en">
      <head>
        <title>{title}</title>
        <meta name="description" content="Next.js 15 + React 19 Application" />
      </head>
      <body className={inter.className}>
        <main>{children}</main>
      </body>
    </html>
  );
}
```

GAMBAR 9
(Implementasi Layout utama Frontend)

Integrasi Firebase sebagai *backend* menyediakan layanan basis data *real-time* dan autentikasi. Integrasi ini mempercepat pengembangan dengan mengurangi kompleksitas infrastruktur *server*, yang merupakan keuntungan signifikan dalam lingkungan pengembangan aplikasi modern. Dengan memanfaatkan Firebase, pengembangan dapat fokus pada fitur inti aplikasi daripada mengelola dan memelihara *server* sendiri.

```
import { initializeApp, getApps } from 'firebase/app';
import { getAuth } from 'firebase/auth';
import { getFirestore } from 'firebase/firestore';
import { getStorage } from 'firebase/storage';

const firebaseConfig = {
  apiKey: process.env.NEXT_PUBLIC_FIREBASE_API_KEY,
  authDomain: process.env.NEXT_PUBLIC_FIREBASE_AUTH_DOMAIN,
  projectId: process.env.NEXT_PUBLIC_FIREBASE_PROJECT_ID,
  storageBucket: process.env.NEXT_PUBLIC_FIREBASE_STORAGE_BUCKET,
  messagingSenderId:
    process.env.NEXT_PUBLIC_FIREBASE_MESSAGING_SENDER_ID,
  appId: process.env.NEXT_PUBLIC_FIREBASE_APP_ID
};

const app = getApps().length === 0 ? initializeApp(firebaseConfig) : getApps()[0];

export const auth = getAuth(app);
export const db = getFirestore(app);
export const storage = getStorage(app);
export default app;
```

GAMBAR 10
(Integrasi Firebase sebagai Backend)

G. Integration and System Testing

Setelah implementasi perangkat lunak, sistem akan melalui fase pengujian integrasi dan sistem untuk memastikan semua komponen berinteraksi dengan baik dan sistem berfungsi sesuai spesifikasi. Dalam fase ini, skenario pengujian seperti proses asesmen, kalkulasi dengan algoritma *Certainty Factor*, dan penyimpanan riwayat pengguna divalidasi secara sistematis.

TABEL 3
(Ringkasan Laporan Pengujian)

Skenario	Deskripsi	Hasil
Autentikasi	Pengujian meliputi <i>log in</i> dan <i>sign up</i> dengan berbagai kombinasi <i>input</i> valid dan tidak valid.	Lulus
Asesmen	Pengujian berfokus pada proses input data asesmen dan penyimpanannya.	Lulus
Global	Pengujian mencakup akses halaman (<i>about dan support</i>) serta perubahan tampilan antarmuka (tema aplikasi).	Lulus
Dashboard Pengguna	Pengujian mencakup navigasi halaman, pembaruan profil, dan validasi input data.	Lulus
Dashboard Admin	Pengujian meliputi skenario pengelolaan data mahasiswa dan asesmen.	Lulus

Laporan pengujian secara keseluruhan menunjukkan bahwa sistem telah berfungsi dengan baik sesuai dengan spesifikasi yang dirancang. Seluruh *use case*, mulai dari proses autentikasi pengguna dan admin, pelaksanaan asesmen stres, hingga fitur navigasi global dan pengelolaan *dashboard*, telah berhasil melewati semua *test case* yang diberikan, baik secara manual maupun otomatis.

H. Operation and Maintenance

Kegiatan operasional sistem terbagi berdasarkan platform yang digunakan. Untuk *hosting* dan *deployment*, sistem mengandalkan Vercel, yang secara otomatis mengelola *server*, *scaling*, dan ketersediaan aplikasi, meminimalkan intervensi manual. Pemantauan kondisi rilis dan analisis log aplikasi untuk penelusuran masalah dilakukan proaktif melalui *dashboard* Vercel. Sementara itu, untuk manajemen data dan autentikasi, sistem memanfaatkan layanan Google Firebase, dengan pengawasan penggunaan basis data Cloud Firestore, performa, dan status layanan autentikasi melalui konsol Firebase.

I. Release

Perilisan aplikasi kepada pengguna akhir dikelola secara efisien melalui Vercel, yang dipilih karena integrasi eratnya dengan Next.js, kerangka kerja pengembangan aplikasi ini. Proses perilisan otomatis dipicu setiap kali pembaruan yang telah teruji digabungkan ke *branch* utama repositori GitHub. Vercel secara mandiri menangani semua langkah teknis, termasuk instalasi dependensi, kompilasi kode, validasi konfigurasi, dan publikasi ke *server* produksi. Setelah semua tahapan berhasil, versi terbaru aplikasi segera dapat diakses pengguna melalui *domain* yang telah diatur, memastikan peningkatan atau perbaikan dirilis dengan cepat dan konsisten.

J. Evaluasi Hasil

Evaluasi sistem dilaksanakan setelah pengembangan dan rilis ke lingkungan produksi. Sembilan responden menguji langsung fitur-fitur utama, termasuk asesmen tingkat stres,

riwayat hasil asesmen, dan pengelolaan akun pengguna. Penilaian responden kemudian dikumpulkan melalui pengisian kuesioner *System Usability Scale* (SUS) secara lengkap, yang bertujuan untuk mengukur pengalaman penggunaan mereka.

TABEL 4
(Hasil Skor Akhir SUS)

No.	Skor Akhir SUS
1	77,5
2	92,5
3	100
4	75
5	90
6	80
7	87,5
8	85
9	77,5

Skor akhir evaluasi *usability* sistem adalah 81,94, yang dihitung dari rata-rata keseluruhan nilai. Skor ini mengindikasikan tingkat kemudahan penggunaan, pemahaman, dan pengoperasian sistem tanpa hambatan yang signifikan, berdasarkan persepsi pengguna. Nilai ini berada pada kisaran atas, yang secara umum menandakan bahwa sistem layak digunakan secara mandiri, dapat dioperasikan dengan cepat, dan menawarkan pengalaman interaksi yang konsisten. Skor di atas 80 menunjukkan bahwa sistem tidak hanya berfungsi dengan lancar, tetapi juga memberikan tingkat kenyamanan dan kepuasan tertentu selama interaksi pengguna.

V. KESIMPULAN

Berdasarkan hasil perancangan, implementasi, dan evaluasi aplikasi web pengukur tingkat stres bagi mahasiswa Universitas Telkom Purwokerto, dapat disimpulkan bahwa sistem ini telah memenuhi tujuan yang ditetapkan. Aplikasi berhasil dibangun menggunakan metode *Waterfall*, dengan alur kerja yang terdokumentasi dan terverifikasi melalui pengujian fungsional dan *usability*.

Seluruh fitur utama, seperti autentikasi, asesmen tingkat stres, pengelolaan hasil, serta *dashboard* untuk pengguna dan admin, telah berhasil diterapkan dan diuji dalam kondisi nyata. Pengujian *Black Box* mengonfirmasi bahwa fungsi berjalan sesuai rancangan, sementara pengujian *usability* dengan *System Usability Scale* menghasilkan skor rata-rata 81,94. Capaian ini menunjukkan bahwa sistem berhasil membantu pengguna mengukur tingkat stres secara mandiri dan sistematis. Untuk penyempurnaan lebih lanjut, disarankan agar evaluasi *usability* melibatkan lebih banyak responden untuk memahami beragam persepsi, dan menerapkan metode *Think-Aloud* selama asesmen untuk mengidentifikasi potensi kebingungan pada pertanyaan.

REFERENSI

- [1] S. Vogel and L. Schwabe, "Learning and memory under stress: implications for the classroom," *npj Science Learn*, vol. 1, no. 1, p. 16011, Jun. 2016, doi: 10.1038/npjscilearn.2016.11.
- [2] N. L. Shankar and C. L. Park, "Effects of stress on students' physical and mental health and academic success," *International Journal of School & Educational Psychology*, vol. 4, no. 1, pp. 5–9, Jan. 2016, doi: 10.1080/21683603.2016.1130532.
- [3] Y. Khallad and F. Jabr, "Effects of perceived social support and family demands on college students' mental well-being: A cross-cultural investigation," *Int J Psychol*, vol. 51, no. 5, pp. 348–355, Oct. 2016, doi: 10.1002/ijop.12177.
- [4] C. R. Hidayat, E. D. S. Mulyani, and I. Agustina, "Sistem Pakar Menganalisis Tingkat Stress Siswa SMA Menggunakan Metode Certainty Factor," presented at the SISITI: Seminar Ilmiah Sistem Informasi dan Teknologi Informasi, 2022, pp. 301–311.
- [5] A. Saravanos and M. X. Curinga, "Simulating the Software Development Lifecycle: The Waterfall Model," 2023, doi: 10.48550/ARXIV.2308.03940.
- [6] M. Narizzano, L. Pulina, A. Tacchella, and S. Vuotto, "Automated Requirements-Based Testing of Black-Box Reactive Systems," in *NASA Formal Methods*, vol. 12229, R. Lee, S. Jha, A. Mavridou, and D. Giannakopoulou, Eds., in Lecture Notes in Computer Science, vol. 12229, Cham: Springer International Publishing, 2020, pp. 153–169. doi: 10.1007/978-3-030-55754-6_9.
- [7] D. M. Brandon, *Software Engineering for Modern Web Applications: Methodologies and Technologies: Methodologies and Technologies*. in IGI Global research collection. Information Science Reference, 2008. [Online]. Available: <https://books.google.co.id/books?id=yDyJ-iH6vpwC>
- [8] R. Pressman, *Software Engineering: A Practitioner's Approach*. McGraw-Hill Education, 2014. [Online]. Available: <https://books.google.co.id/books?id=dXlzCgAAQBAJ>
- [9] I. Sommerville, *Software Engineering, Global Edition*. Pearson Education, 2016. [Online]. Available: https://books.google.co.id/books?id=W_LjCwAAQBAJ
- [10] M. K. Anik Andriani, *Pemrograman Sistem Pakar*. Media Pressindo, 2016. [Online]. Available: <https://books.google.co.id/books?id=wyrJEAAAQBAJ>
- [11] J. Chen *et al.*, "Wireframe-based UI Design Search through Image Autoencoder," *ACM Trans. Softw. Eng. Methodol.*, vol. 29, no. 3, pp. 1–31, Jul. 2020, doi: 10.1145/3391613.
- [12] M. Madsen, O. Lhoták, and F. Tip, "A Semantics for the Essence of React," *LIPIcs, Volume 166, ECOOP 2020*, vol. 166, p. 12:1–12:26, 2020, doi: 10.4230/LIPICS.ECOOP.2020.12.
- [13] Md. S. A. Khan, A. R. Farabi, and A. Iqbal, "What Do Firebase Developers Discuss About? An Empirical

Study on Stack Overflow Posts,” in *Proceedings of the 9th International Conference on Networking, Systems and Security*, Cox’s Bazar Bangladesh: ACM, Dec. 2022, pp. 63–74. doi: 10.1145/3569551.3569558.

- [14] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*. in Addison-Wesley object technology series. Addison-Wesley, 2005. [Online]. Available: <https://books.google.co.id/books?id=BqFQAAAAMA AJ>
- [15] B. Albert and T. Tullis, *Measuring the User Experience: Collecting, Analyzing, and Presenting*

Usability Metrics. in Interactive Technologies. Elsevier Science, 2013. [Online]. Available: <https://books.google.co.id/books?id=bPhLeMBLEkA C>

- [16] B. B. Agarwal, S. P. Tayal, and M. Gupta, *Software Engineering and Testing*. in Computer science series. Jones & Bartlett Learning, 2010. [Online]. Available: <https://books.google.co.id/books?id=ZoF06z4dhQ4C>

