ANTIVIRUS: Jurnal Ilmiah Teknik Informatika (p – ISSN: 1978 – 5232; e – ISSN: 2527 – 337X)
Vol. 19 No. 1 Mei 2025, pp. 67 – 82

PENGEMBANGAN WEB MANAJEMEN TUGAS DAILIDO MENGGUNAKAN TEKNOLOGI MERN STACK

Diterima Redaksi:28 Maret 2025; Revisi Akhir: 7 Mei 2025; Diterbitkan Online: 30 Mei 2025

Amaliyah[1)], Apriade Voutama[2)]
[1, 2)] Sistem Informasi, Fakultas Ilmu Komputer Universitas Singaperbangsa Karawang
[1, 2)] Jalan HS.Ronggo Waluyo,Puseurjaya,Telukjambe Timur, Karawang,Jawa Barat 41361
e-mail: amaliyahamel0304@gmail.com[1)], apriade.voutama@staff.unsika.ac.id[2)]

***Abstrak*:** *Penelitian ini mengembangkan Dailido, aplikasi web manajemen tugas berbasis MERN Stack (MongoDB, ExpressJS, ReactJS, NodeJS) untuk menyediakan solusi sederhana dan efektif dalam mengatur tugas harian. Dengan menerapkan Rapid Application Development (RAD), proses pengembangan dapat dipercepat, sementara Black Box Testing dan Single Ease Question (SEQ) memvalidasi fungsionalitas dan kemudahan penggunaan. Hasilnya, Dailido berhasil mengimplementasikan operasi dasar (buat, baca, edit, hapus tugas) dengan antarmuka intuitif, dimana pengujian membuktikan seluruh fitur bekerja optimal tanpa kesalahan. Penelitian ini menyimpulkan bahwa pendekatan MERN Stack dan RAD cocok untuk pengembangan aplikasi manajemen tugas yang ringan namun sesuai kebutuhan pengguna.*
.
Kata Kunci— Aplikasi manajemen tugas, Black Box Testing, Dailido, MERN Stack, Rapid Application Development (RAD)

***Abstract*:** *This research develops Dailido, a task management web application based on MERN Stack (MongoDB, ExpressJS, ReactJS, NodeJS) to provide a simple and effective solution to organize daily tasks. By applying Rapid Application Development (RAD), the development process was accelerated, while Black Box Testing and Single Ease Question (SEQ) validated the functionality and ease of use. As a result, Dailido successfully implemented basic operations (create, read, edit, delete tasks) with an intuitive interface, where testing proved all features work optimally without errors. This research concludes that MERN Stack and RAD approaches are suitable for the development of a lightweight yet user-friendly task management application.*

Keywords— Black Box Testing, Dailido, MERN Stack, Rapid Application Development (RAD), task management application

I. PENDAHULUAN

Perkembangan teknologi sangat mempengaruhi dan memberikan manfaat yang besar di berbagai aspek kehidupan [1]. Salah satu teknologi yang paling pesat berkembang adalah Internet. Berkat internet, aktivitas manusia kini tidak terikat oleh batasan ruang dan waktu. [2]. Dunia digital yang serba cepat ini, menuntut setiap orang untuk memiliki produktivitas yang tinggi. Pengelolaan waktu dan skala prioritas menjadi sangat penting dalam meningkatkan produktivitas [3].

Bagi beberapa orang mengatur banyak tugas bisa menjadi beban yang berat dan penuh kendala, salah satu permasalahan umum yang sering dijumpai adalah siswa sering terlambat mengumpulkan tugas di sekolah [4]. Contoh nyata lainnya terjadi pada para *freelancer*, banyaknya pekerjaan terkadang membuat *freelancer* tidak bisa membatasi antara rumah dan pekerjaan, mereka sering menghadapi kesulitan untuk menentukan prioritas pekerjaan yang harus diselesaikan terlebih dahulu [5]. Karenanya, kehadiran aplikasi manajemen tugas atau *to-do list* dapat mempermudah dalam mencatat, menyusun, dan melacak riwayat tugas-tugas yang perlu dikerjakan [3].

Banyak individu masih menggunakan cara tradisional untuk membuat daftar tugas, seperti mencatatnya secara manual di atas kertas. Kemudian sebagai pengecekan batas waktu tugas juga harus

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

dilakukan secara berkala yang membuat *to-do list* menjadi kurang efisien [6]. Berdasarkan masalah-masalah tersebut dibuatlah sebuah aplikasi berbasis *website* yang berfungsi untuk memanajemen tugas pengguna bernama Dailido. Pengembangan aplikasi ini memanfaatkan teknologi MERN Stack, menggunakan model RAD, dan metode pengujian *Black Box Testing* dan *Single Ease Question* (SEQ).

Tujuan dibuatnya aplikasi Dailido adalah untuk membantu pengguna menjadi lebih mudah dalam mengorganisir tugas-tugasnya. Dailido dapat melacak tugas apa saja yang sudah dikerjakan dan melihat daftar riwayat tugas pengguna. Dengan adanya aplikasi ini diharapkan pengguna dapat meningkatkan produktifitas dan efisiensi dalam pengerjaan tugas.

Pengembangan aplikasi manajemen tugas yang sudah ada sebelumnya diantaranya adalah Aplikasi Manajemen Tugas berbasis Android pada CV. Cheleron Production oleh Iqra Ilhamsyah [7], namun penelitian Iqra Ilhamsyah berbasis Android dan diterapkan dengan bahasa pemrograman Java, sedangkan aplikasi Dailido berbasis *webiste* dan menggunakan MERN Stack sehingga penggunaan aplikasi ini dapat digunakan diberbagai *device* karena tampilannya yang responsif. Selain itu target pengguna Aplikasi Dailido lebih luas dan dapat gunakan oleh setiap individu maupun suatu organisasi, sementara pada penelitian Iqra Ilhamsyah aplikasi ditunjukkan untuk karyawan CV. Cheleron Production.

MERN Stack telah digunakan dalam beberapa studi terdahulu, termasuk riset yang dikerjakan oleh Ade Lintang Permono, penerapan MERN Stack pada Aplikasi Jadwal Plus sebagai Sistem Manajemen Penjadwalan [8] persamaan kedua aplikasi adalah penggunaan *tech* Stack yang sama sebagai teknologi utama pengembangannya. Namun penelitian ini berbeda, karena fokus pada *task management* dasar (CRUD tugas dan autentikasi pengguna) tanpa fitur tambahan yang rumit, dan pendekatan minimalis yang disengaja untuk pengguna awam. Kontribusi utama penelitian ini adalah pembuktian bahwa solusi *ultra-simplified* berbasis MERN Stack dapat memenuhi kebutuhan manajemen tugas harian tanpa kompleksitas sistem terdahulu.

II. TINJAUAN PUSTAKA

A. *Website Dailido*

Perencanaan yang terstruktur akan sangat memudahkan dan mempercepat pengerjaan banyak tugas. Aplikasi Dailido adalah aplikasi berbasis *website* yang secara khusus dikembangkan untuk manajemen tugas-tugas pengguna, disertai dengan fitur daftar dan *login* memastikan keamanan pengguna. Aplikasi Dailido juga dapat digunakan untuk *tracking* bagi pengguna dalam proses pengerjaan tugas-tugas.

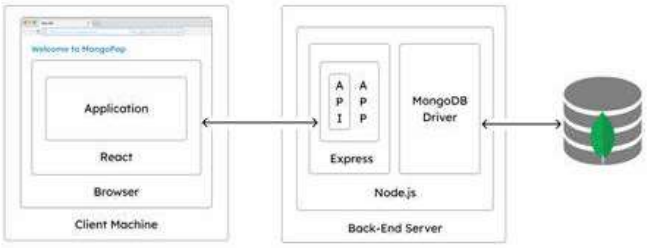
B. *Manajemen Tugas*

Pengelolaan tugas mencakup berbagai langkah, seperti perencanaan, pengujian, pemantauan, dan pelaporan. Di tengah perkembangan teknologi saat ini, manajemen tugas merupakan komponen krusial. Pada dasarnya, pengelolaan tugas merupakan kegiatan yang bertujuan untuk mengatur pekerjaan secara efisien demi mencapai target yang telah ditetapkan [9].

C. *MERN* Stack

MERN Stack merupakan salah satu *tech* Stack yang sangat populer. MERN Stack menggunakan bahasa pemrograman jJavascript dan terdiri dari MongoDB, ExpressJS, ReactJS, dan NodeJS yang digunakan dalam pembangunan aplikasi *website*. *Frontend* aplikasi menggunakan ReactJS yang dapat memberikan *user interface* yang interaktif dan dapat digunakan di berbagai ukuran layar [10]. Kemudian dalam pengelolaan basis data, penelitian ini menggunakan MongoDB sebagai NoSQL yang bersifat fleksibel, sementara itu untuk mempermudah pengembangan API digunakan ExpressJS yang merupakan *framework* NodeJS. NodeJS adalah lingkungan *runtime* yang berfungsi untuk menjalankan Javascript di *server* [11].

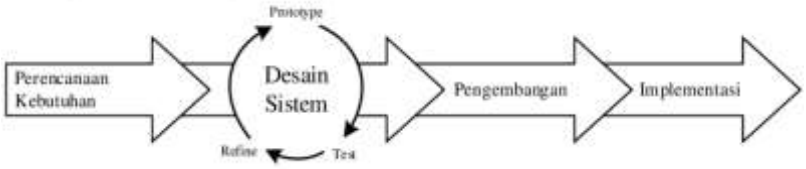
DOI: https://doi.org/10.35457/antivirus.v19i1.4435



Gambar 1. Pengembangan Website dengan Teknologi MERN Stack

D. Rapid Application Development (RAD)

Ada berbagai macam metode yang dapat dipilih dalam pengembangan *website*, salah satunya metode RAD yang merupakan pendekatan dalam membuat perangkat lunak yang fokus pada penyelesaian siklus pengembangan secara cepat dan terstruktur. Program yang dibuat pertama kali diuji dan jika terjadi kesalahan, kemudian diperbaiki kembali sehingga program berjalan sesuai yang direncanakan [12]. Hasil dari pengembangan sistem dengan metode RAD berkualitas baik dan biaya pengembangannya relatif lebih murah [13].



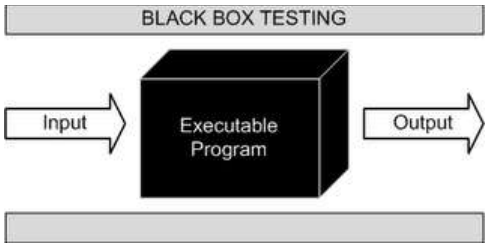
Gambar 2. Tahapan Metode Rapid Application Development

Pengembangan dengan Metode RAD secara umum melibatkan tahapan-tahapan berikut [14]:

1. Rencana Kebutuhan yaitu melakukan identifikasi terkait kebutuhan yang diperlukan dalam proses pengembangan dan analisis tujuan serta tahapan untuk mencapainya.
2. Proses Desain Sistem adalah pada tahap dimana pengembang dan pengguna melakukan kolaborasi untuk membangun sistem berdasarkan kebutuhan yang sudah ditentukan. Pengguna dapat memberikan masukkan terkait kekurangan dan kemudian dilakukan evaluasi.
3. Pengembangan aplikasi hingga versi final, proses mencangkup pengembangan terus menerus dan penyesuaian berdasarkan masukkan pengguna.
4. Implementasi tahap pengujian terakhir dari aplikasi yang sudah disetujui, pengguna memberikan tanggapan dan persetujuan sebelum sistem digunakan.

E. Black Box Testing

Ketika tahap pengembangan telah menghasilkan sebuah sistem sementara/prototipe maka akan dilakukan tahap pengujian (*testing*) dari sistem tersebut [15]. Proses *testing* salah satunya adalah *black box testing* yang dilakukan dengan mengutamakan spesifikasi aplikasi yang dikembangkan, yaitu dengan menganalisis dan mengumpulkan input dan output yang diharapkan serta kesesuaian terhadap spesifikasi yang sudah ditentukan [16].



Gambar 3. Ilustrasi Penerapan Black Box Testing

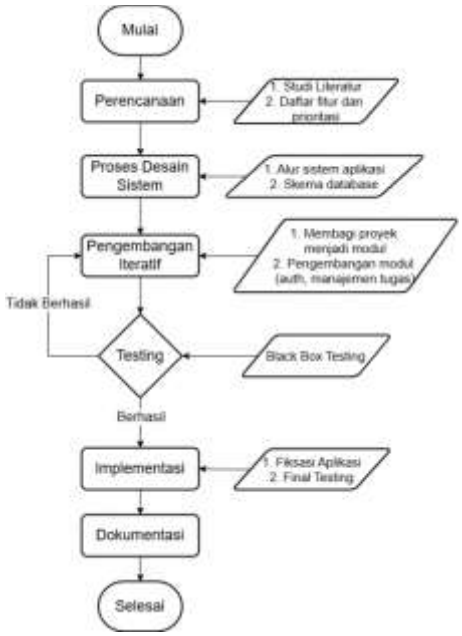
DOI: https://doi.org/10.35457/antivirus.v19i1.4435

F. Single Ease Question (SEQ)

Usability testing saat ini menerapkan berbagai macam teknik, termasuk *Single Ease Question* (SEQ). Teknik ini menggunakan kuesioner singkat yang diisi partisipan setelah menyelesaikan suatu tugas selama pengujian. Penggunaan SEQ memungkinkan fasilitator memperoleh umpan balik secara instan terkait kesulitan yang dialami responden dalam menyelesaikan tugas. Keunggulan lainnya, partisipan dapat memberikan evaluasi dengan lebih akurat karena penilaian dilakukan tepat setelah tugas selesai. Skor tugas diukur menggunakan skala 1–7, di mana 1 menandakan tingkat kesulitan tertinggi dan 7 menunjukkan kemudahan maksimal. Jika rata-rata SEQ ≤ 5, tugas tersebut tergolong sulit dan berada di bawah standar [17].

III. METODE PENELITIAN

Penelitian ini menerapkan pendekatan pengembangan berbasis *Rapid Application Development* (RAD). Pemilihan metode RAD dikarenakan metode ini dapat membuat proses pengembangan menjadi lebih cepat dan iteratif, sehingga dapat menyesuaikan kebutuhan pengguna dan melakukan evaluasi selama proses pengembangan.



Gambar 4. Alur Penelitian dengan Metode RAD

Pada gambar di atas merupakan tahapan-tahapan dalam proses pengembangan aplikasi Dailido dengan menggunakan metode RAD.

A. Tahap Perencanaan (Requirements Planning)

Tahapan pertama dalam pengembangan aplikasi Dailido adalah perencanaan kebutuhan. Melakukan identifikasi kebutuhan fungsional dan non-fungsional aplikasi berdasarkan studi literatur yang sudah dilakukan. Kemudian merancang fitur-fitur utama aplikasi berdasarkan analisis kebutuhan umum untuk aplikasi manajemen tugas, seperti otentikasi pengguna (*register* dan *login*), manajemen tugas, dan manajemen pengguna.

B. Tahap Perancangan Desain Sistem

Setelah identifikasi kebutuhan sistem selesai, langkah berikutnya adalah merancang sistem. Pada fase ini, diagram alur (*flowchart)* dibuat untuk menjelaskan alur kerja aplikasi secara keseluruhan. Arsitektur sistem disiapkan dengan membagi aplikasi menjadi tiga komponen utama, yaitu *frontend* menggunakan ReactJS, *backend* menggunakan NodeJS dan ExpressJS, serta *database* menggunakan MongoDB. Sebelum pembuatan *database* di MongoDB akan dirancang desain *database* yang akan mempermudah pembuatan struktur basis data dari aplikasi Dailido.

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

C. Tahap Pengembangan Iteratif (Rapid Construction)

Tahap pengembangan dilakukan secara iteratif dengan membagi proyek menjadi modul-modul kecil, seperti modul otentikasi pengguna, modul manajemen tugas, dan manajemen pengguna. Setiap modul dikembangkan secara terpisah. Untuk mempercepat pengembangan, digunakan *library* dan *tools* pendukung seperti Tailwind CSS untuk *styling* di *frontend*, JWT (JSON Web Tokens) untuk otentikasi. Proses pengembangan dilakukan secara mandiri dengan fokus pada penyelesaian fitur-fitur inti terlebih dahulu.

D. Tahap Testing dan Evaluasi (Cutover)

Pengujian sistem dilakukan melalui dua pendekatan utama:

1) Black Box Testing

Pengujian fungsional menggunakan metode *black box* dengan menyusun 12 skenario uji berdasarkan kebutuhan sistem. Tabel hasil *black box testing* mencakup: a) skenario uji coba, b) fungsi yang diuji, c) ekspektasi hasil, d) hasil actual, dan e) kesimpulan (Valid/Invalid).

2) Single Ease Question (SEQ)

Pengujian menggunakan *Single Ease Question* (SEQ) setelah responden menyelesaikan 7 tugas utama: 1) registrasi akun, 2) login akun, 3) membuat tugas baru, 4) menandai tugas selesai, 5) menghapus tugas, 6) mengganti data profil, 7) keluar akun

Prosedur pengujian SEQ menggunakan responden 5 mahasiswa Sistem Informasi (semester 4-6), dengan kriteria responden, yaitu: a) pengguna aktif aplikasi manajemen tugas (minimal 3x/bulan), dan b) tidak pernah menggunakan sistem ini sebelumnya. Pertanyaan SEQ yang diajukan, "Seberapa mudah Anda menyelesaikan tugas ini?", dengan skala 1-7 (1 = Sangat Sulit, 7 = Sangat Mudah). Analisis data dilakukan dengan menghitung rata-rata skor SEQ tiap tugas. Kategori hasil skor SEQ:

a. ≥6.5: Sangat Mudah
b. 5.0-6.4: Mudah
c. ≤4.9: Perlu Perbaikan

E. Tahap Implementasi dan Dokumentasi

Tahap terakhir adalah implementasi aplikasi Dailido, *frontend* dan *backend* dikembangkan secara terpisah, dengan *backend* dijalankan secara lokal. *Database* dikonfigurasi menggunakan MongoDB untuk memastikan integrasi dan keamanan data. Setelah seluruh fitur selesai dikerjakan, dilakukan uji coba komprehensif guna memverifikasi bahwa setiap bagian aplikasi bekerja secara optimal.

IV. HASIL DAN PEMBAHASAN

A. Analisis Kebutuhan

Berikut ini daftar kebutuhan sistem untuk aplikasi manajemen tugas:

Tabel 1. Spesifikasi Kebutuhan (Requirements Specification Table)

Analisis Kebutuhan Fungsional	
No.	Diinginkan sebuah sistem yang mampu
1.	Pengguna baru dapat membuat akun dengan mengisi formulir registrasi.
2.	Pengguna dapat masuk ke aplikasi menggunakan *email* dan *password*.
3.	Pengguna memiliki kemampuan untuk menambahkan tugas baru.
4.	Pengguna bisa memeriksa daftar pekerjaan/tugas yang sudah dibuat.
5.	Pengguna dapat mengedit dan mengubah status tugas.
6.	Pengguna bisa menghapus pekerjaan yang sudah tidak dibutuhkan.
7.	Pengguna dapat melihat informasi profil mereka, seperti nama, *email*, dan foto profil.
8.	Pengguna dapat mengubah informasi profil, seperti nama dan foto profil.
9.	Pengguna dapat mengganti *password* akun mereka.
10.	Pengguna dapat keluar dari aplikasi dengan aman.
Analisis Kebutuhan Non-Fungsional	
No.	Diinginkan sebuah sistem yang mampu
1.	Data pengguna (seperti *password*) harus dienkripsi menggunakan metode yang aman (misalnya, *hashing* dengan *bcrypt*).
2.	Antarmuka pengguna perlu dirancang sederhana dan intuitif agar dapat digunakan dengan mudah,

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

termasuk oleh orang tanpa pengetahuan teknis.
3. Aplikasi harus responsif dan dapat digunakan di berbagai ukuran *device*.

Berikut ini daftar kebutuhan sistem untuk aplikasi manajemen tugas:

Tabel 2. Fitur-Fitur Utama Pengguna

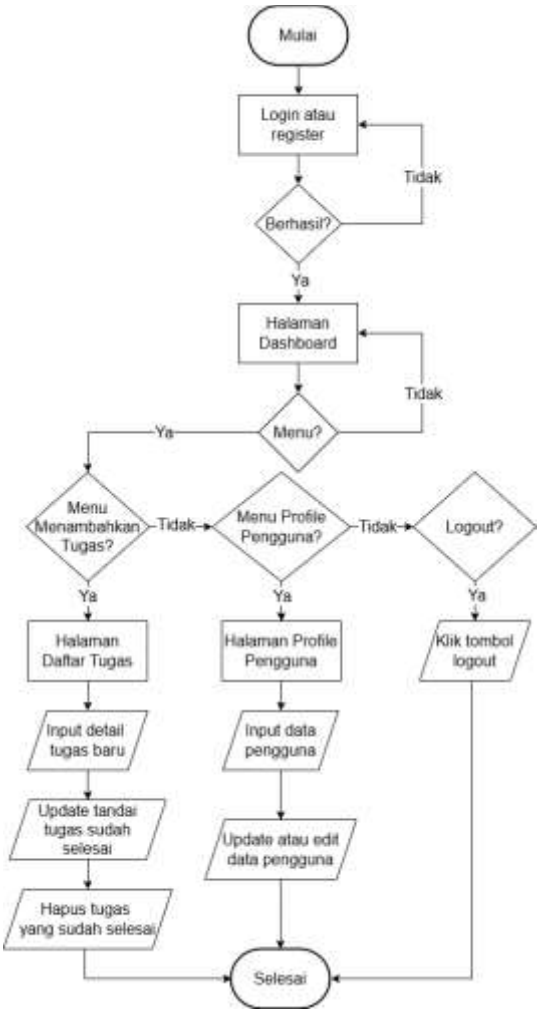
No.	Fitur
1.	Daftar
2.	*Login*
3.	*Homepage*
3.	Melihat Daftar Tugas
4.	Menambahkan Tugas
5.	Mengedit Status Tugas
6.	Menghapus Tugas
7.	Melihat *Profile* Pengguna
8.	Menambahkan Data pada *Profile* Pengguna
9.	Mengedit Data pada *Profile* Pengguna

B. Perancangan Desain Sistem

Langkah berikutnya adalah fase perancangan sistem, yang akan menjadi pedoman dalam pengembangan aplikasi Dailido pada tahap-tahap selanjutnya.

1) Diagram Alir Sistem Aplikasi Dailido

Berikut adalah gambar diagram alir yang menjelaskan proses alur sistem pengelolaan tugas Dailido.

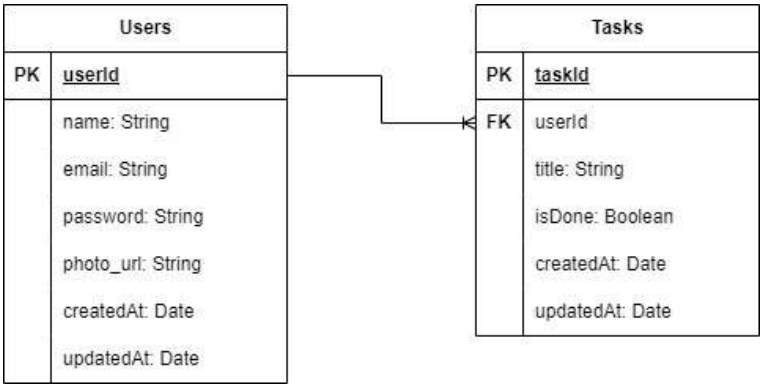


Gambar 5. Diagram Alir Sistem (Flowchart)

2) Desain Database Sistem Aplikasi Dailido

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

Diagram ini merepresentasikan desain *database* MongoDB untuk aplikasi Dailido. Notasi tabel-kolom dipilih untuk mempermudah pemahaman hubungan antar data.



Gambar 6. Desain Database Aplikasi Dailido

C. Pengembangan Aplikasi Menggunakan MERN Stack

Sistem Dailido dikembangkan dengan arsitektur MERN Stack (MongoDB, ExpressJS, ReactJS, NodeJS). Implementasi tiap komponen dijelaskan sebagai berikut:

1) Backend Layer (NodeJS & ExpressJS)

a) Runtime Environment:
NodeJS v20.18.0 (Non-LTS).

b) Framework:
Framework ExpressJS v4.21.1 diimplementasikan dengan:
1. Middleware standar untuk keamanan (*cors*) dan observabilitas (*morgan*)
2. Arsitektur modular dengan pemisahan *concern* (*routes-controllers-models*)
3. Dokumentasi API otomatis menggunakan *Swagger OpenAPI* 3.0

c) Kode App.js sebagai inti *backend* Aplikasi Dailido:

```
const express = require('express');
const cors = require('cors');
const swaggerJsDoc = require('swagger-jsdoc');
const swaggerUi = require('swagger-ui-express');
const connectDB = require('./config/database');
const { port } = require('./config/env');
const errorHandler = require('./middleware/errorHandler');
const userRoutes = require('./routes/userRoutes');
const taskRoutes = require('./routes/taskRoutes');
const authRoutes = require('./routes/authRoutes');
const morgan = require('morgan');
require("dotenv").config();

const app = express();

connectDB();

app.use(morgan('dev'));

app.use(cors());
app.use(express.json());

const swaggerOptions = {
    definition: {
        openapi: '3.0.0',
        info: {
            title: 'DailiDo Task Management API',
            version: '1.0.0',
            description: 'API documentation for Task Management System'
        },
```

Gambar 7. Kode file App.js pada backend

2) Database Layer (MongoDB)

Implementasi kode untuk model data sistem:

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

a) Users (Pengguna)

Data pengguna memerlukan nama, email, password, dan url foto. Pada data nama, email dan password bersifat wajib diisi (*required*).

```
const userSchema = new mongoose.Schema({
    name: {
        type: String,
        required: [true, 'Name is required'],
        trim: true
    },
    email: {
        type: String,
        required: [true, 'Email is required'],
        unique: true,
        trim: true,
        lowercase: true,
        match: [/^\w+([.-]?\w+)*@\w+([.-]?\w+)*(\.\w{2,3})+$/, 'Please enter a valid email']
    },
    password: {
        type: String,
        required: [true, 'Password is required'],
        minlength: [6, 'Password must be at least 6 characters']
    },
    photo_url: {
        type: String,
        default: ''
    }
}, {
    timestamps: true
});
```

Gambar 8. Kode User.js

b) Tasks (Tugas)

Model data tugas terdiri dari judul, status selesai, dan id user. Setiap tugas diharuskan memiliki refrensi dari pengguna yang membuat tugas tersebut, kemudian judul tugas juga wajib diisi.

```
const taskSchema = new mongoose.Schema({
    title: {
        type: String,
        required: [true, 'Title is required'],
        trim: true
    },
    isDone: {
        type: Boolean,
        default: false
    },
    userId: {
        type: mongoose.Schema.Types.ObjectId,
        ref: 'User',
        required: [true, 'User ID is required']
    }
}, {
    timestamps: true
});
```

Gambar 9. Kode Task.js

3) Frontend Layer (ReactJS)

Aplikasi frontend dibangun dengan: a) *Routing*: React Router v6 untuk navigasi SPA, b) *State Management*: Native Hooks (*useState, useContext*), dan c) integrasi API: Axios untuk komunikasi dengan *backend*. Deteksi penggunaan ReactJS pada aplikasi Dailido dengan menggunakan *extension*.

Gambar 10. Hasil deteksi menggunakan ReactJS pada Web

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

4) Integrasi Sistem

a) Implementasi API

Frontend dan *backend* terintegrasi melalui RESTful API dengan *endpoint* utama berikut:

Tabel 3. Endpoint Sistem Dailido

Endpoint	Metode	Deskripsi
/api/users/register	POST	Pendaftaran pengguna baru
/api/users/login	POST	Login pengguna
/api/users/profile	GET	Mendapatkan profil pengguna
/api/users/profile	PUT	Memperbarui profil pengguna
/api/tasks	GET	Mendapatkan daftar semua tugas
/api/tasks	POST	Membuat tugas baru
/api/tasks	PATCH	Memperbarui data tugas

D. Hasil dan Tampilan Antarmuka Pengguna

Dalam proses pengembangan aplikasi Dailido terdapat tiga modul terpisah yang dikembangkan yaitu modul otentikasi (*register* dan *login*), modul manajemen tugas, dan modul manajemen pengguna. Di bawah ini adalah antarmuka pengguna untuk berbagai fitur yang tersedia di aplikasi Dailido.

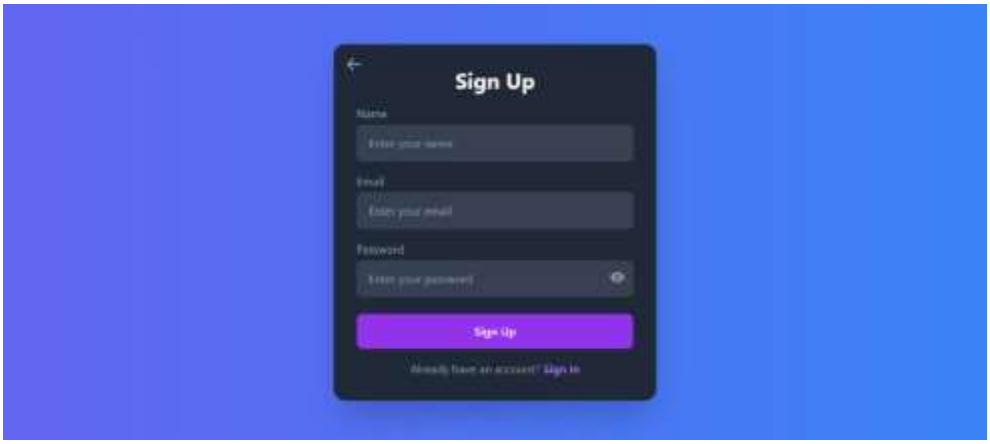
1) Tampilan Homepage

Halaman Beranda ditampilkan saat paling awal pengguna menggunakan aplikasi, pada halaman ini terdapat tombol *sign in* (masuk) dan *sign up* (daftar).

Gambar 11. Tampilan Homepage

2) Tampilan Halaman Daftar/Register

Bagi yang belum memiliki akun, perlu mendaftar terlebih dulu melalui halaman registrasi.



Gambar 12. Tampilan Halaman Daftar (Sign Up)

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

3) Tampilan Halaman Login

Pengguna yang sudah mendaftarkan akunnya dapat masuk/*login* pada halaman *login* dengan memasukkan *email* dan *password*.

Gambar 13. Tampilan Halaman Login

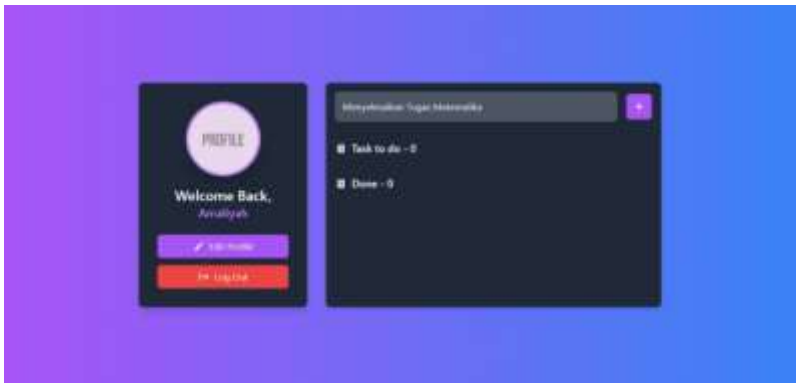
4) Tampilan Halaman Daftar Tugas

Setelah *login* pengguna akan diarahkan pada halaman yang menampilkan daftar tugas dan *profile* pengguna, pengguna dapat menambahkan tugas, mengedit tugas, dan menghapus tugas pada halaman ini, selanjutnya terdapat juga tombol edit *profile* dan *logout*.

Gambar 14. Tampilan Halaman Daftar Tugas

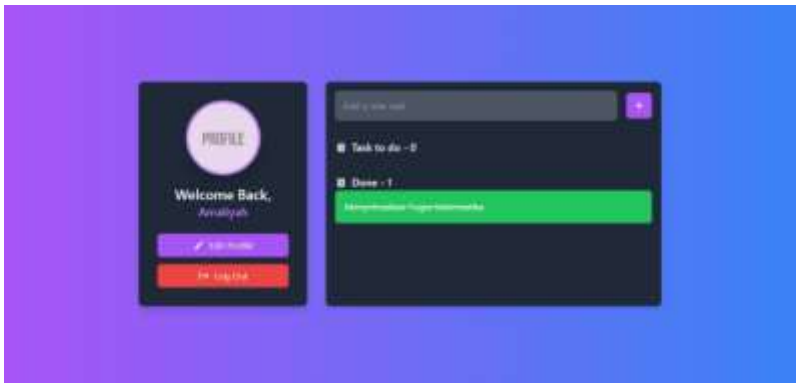
5) Tampilan Proses Menambahkan dan Menyelesaikan Tugas

Pengguna dapat membuat tugas baru dengan menulis judul tugas dan klik tombol tambah, kemudian tugas ditandai sebagai selesai dengan klik centang pada sisi kiri tugas.



Gambar 15. Tampilan Menambahkan Tugas

DOI: https://doi.org/10.35457/antivirus.v19i1.4435



Gambar 16. Tampilan Tugas Selesai

6) Tampilan Proses Hapus Tugas

Daftar tugas yang sudah dibuat dapat dihapus jika terjadi kesalahan, yaitu dengan klik silang pada sebelah kiri judul tugas.

Gambar 17. Sebelum Tugas Dihapus

Gambar 18. Setelah Tugas Dihapus

7) Tampilan Halaman Edit Profile Pengguna

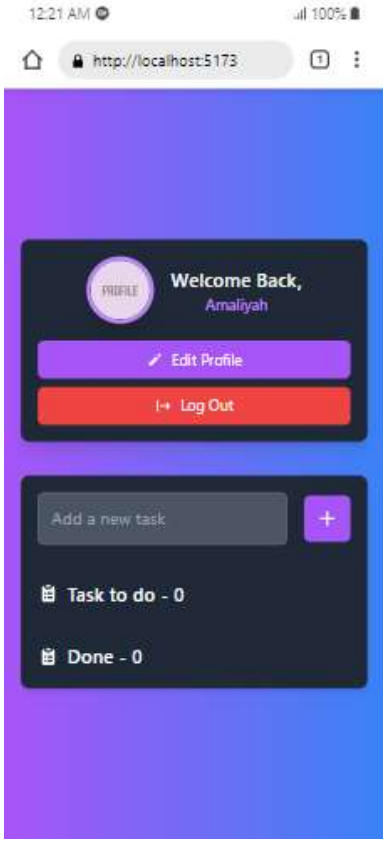
Profile pengguna dapat diperbaharui melalui halaman edit *profile*, adapun data-data yang bisa diedit adalah foto profil, nama, *email*, dan *password*.

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

Gambar 19. Tampilan Halaman Edit Profile

8) Tampilan Halaman Responsif di Device Mobile

Halaman-halaman sistem manajemen tugas Dailido dapat menyesuaikan berbagai ukuran *device* (responsif), salah satunya adalah pada halaman daftar tugas, berikut tampilannya.



Gambar 20. Tampilan Halaman pada Versi Mobile

E. Testing

1) Black Box Testing

Pengujian fungsional menggunakan metode *black box testing* terhadap aplikasi Dailido menunjukkan hasil yang sangat positif. Dari 12 skenario pengujian, seluruhnya berhasil dilewati dengan sempurna, mencapai tingkat keberhasilan 100%. Hasil ini mengindikasikan bahwa semua komponen fungsional aplikasi telah bekerja sesuai dengan spesifikasi dan kebutuhan yang telah ditetapkan.

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

Tabel 4. Black Box Testing Sistem Dailido

Hasil Pengujian Dengan Black Box Testing Fungsional					
No	**Skenario Uji Coba**	**Uji Fungsi**	**Ekspektasi Hasil**	**Hasil Uji**	**Kesimpulan**
1.	Pengguna dapat mendaftarkan akun melalui halaman *register*.		Pengguna berhasil mendaftarkan akun dan data masuk ke *database*.		Valid
2.	Pengguna bisa *login* dengan memakai akun yang telah terdaftar.		Pengguna berhasil *login* dan masuk ke halaman daftar tugas.		Valid
3.	Pengguna tidak dapat *login* jika *email*/*password* yang dimasukkan tidak valid.		Pengguna gagal *login* dan muncul *pop-up* peringatan.		Valid
4.	Pengguna dapat menambahkan tugas baru.		Pengguna berhasil menambahkan tugas baru		Valid
5.	Pengguna bisa memeriksa daftar pekerjaan atau tugas yang telah dibuat.		Semua pekerjaan yang telah dibuat oleh pengguna berhasil ditampilkan.		Valid
6.	Pengguna dapat mengedit dan mengubah status tugas.		Tugas berhasil ditandai sebagai selesai dan berpindah ke bagian *done*.		Valid
7.	Pengguna dapat menghapus tugas yang ingin dihapus.		Tugas berhasil dihapus dan menghilang dari daftar tugas.		Valid
8.	Pengguna dapat melihat informasi profil mereka, seperti nama, *email*, dan foto profil.		Halaman edit *profile* menampilkan data dari pengguna pada saat ini.		Valid

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

Hasil Pengujian Dengan Black Box Testing Fungsional					
No	**Skenario Uji Coba**	**Uji Fungsi**	**Ekspektasi Hasil**	**Hasil Uji**	**Kesimpulan**
9.	Pengguna dapat mengubah informasi profil, seperti nama dan foto profil.		Berhasil update data pengguna dan data baru disimpan di *database*.		Valid
10.	Pengguna dapat keluar dari aplikasi dengan aman dengan klik tombol *logout*.		Setelah klik tombol *logout* pengguna keluar dan masuk *homepage*.		Valid

Hasil Pengujian dengan Blackbox Testing Non-Fungsional					
No	**Kasus Uji Coba**	**Uji Fungsi**	**Ekspektasi Hasil**	**Hasil Uji**	**Kesimpulan**
11.	*Password* pengguna telah diterapkan *hashing*.		Data *password* di *database* dalam bentuk simbol acak (*hashing)*		Valid
12.	Pengguna dapat menggunakan sistem diberbagai *device*		Melalui laptop ataupun *mobile* tampilan website tetap bisa digunakan		Valid

2) Single Ease Question (SEQ)

Pengujian dilakukan melalui usability testing dengan pendekatan *Single Ease Question*. Berikut merupakan hasil evaluasi SEQ dari para responden.

a) Profil Responden
   1) 5 mahasiswa Sistem Informasi (semester 4-6)
   2) Pengguna aktif aplikasi manajemen tugas (≥3x/bulan)
   3) Pengguna baru sistem Dailido

b) Tugas yang harus dikerjakan oleh responden
   1) Registrasi akun
   2) Login akun
   3) Membuat tugas baru
   4) Menandai tugas selesai
   5) Menghapus tugas
   6) Mengganti data profil
   7) Keluar akun

c) Hasil Pengujian

Berikut adalah tabel skor-skor yang diberikan oleh responden. Penilaian kemudahan menggunakan skala SEQ (1-7) dari 1 paling sulit hingga 7 paling mudah.

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

Tabel 5. Hasil Skor SEQ Sistem Dailido

Tugas	Skor SEQ yang diberikan oleh Responden					Rata-Rata
	Responden 1	Responden 2	Responden 3	Responden 4	Responden 5	
Tugas 1	6	6,5	5	5	5,5	5,6
Tugas 2	6,5	6,5	5,5	5	6	5,9
Tugas 3	7	7	6	6	6,5	6,5
Tugas 4	7	6,5	7	6,5	6,5	6,7
Tugas 5	7	7	7	6,5	7	6,9
Tugas 6	5	4,5	5,5	5	5,5	5,1
Tugas 7	7	7	6,5	7	6,5	6,8
Rata-Rata Keseluruhan						5,43

Responden menyelesaikan total 7 tugas, dengan skor rata-rata SEQ mencapai 5,43 yang berada di atas angka 5. Hasil ini menunjukkan bahwa aplikasi Dailido dalam studi ini memiliki *usability* yang tergolong tinggi dan memberikan kemudahan dalam penggunaannya.

V. KESIMPULAN DAN SARAN

Berdasarkan serangkaian pengembangan dan pengujian, penelitian ini berhasil menciptakan Dailido, sebuah aplikasi manajemen tugas berbasis MERN Stack yang dirancang dengan antarmuka minimalis dan mudah digunakan. Hasil pengujian *Black Box Testing* terhadap 12 skenario utama menunjukkan tingkat keberhasilan 100%, membuktikan bahwa seluruh fitur inti seperti registrasi pengguna, dan pembuatan tugas berfungsi sesuai harapan. Selain itu, pengujian *usability* menggunakan *Single Ease Question* (SEQ) menghasilkan skor rata-rata 5,43, mengindikasikan bahwa aplikasi ini tergolong mudah digunakan. Keunggulan MERN Stack turut berkontribusi pada responsivitas aplikasi di berbagai perangkat serta kemudahan dalam pengembangan fitur.

Meski demikian, penelitian ini memiliki beberapa keterbatasan, seperti belum adanya fitur penentuan prioritas tugas, sistem notifikasi, dan *deployment* ke server publik. Pengujian juga baru dilakukan dalam lingkup terbatas, baik dari segi jumlah responden maupun lingkungan pengujian. Untuk penelitian selanjutnya, rekomendasi pengembangan meliputi penambahan fitur notifikasi dan *reminder*, integrasi sistem prioritas tugas, serta perluasan pengujian ke lebih banyak pengguna dengan beragam latar belakang. Dengan penyempurnaan tersebut, Dailido diharapkan dapat menjadi solusi manajemen tugas yang lebih komprehensif, efisien, dan berdampak positif pada produktivitas pengguna.

DAFTAR PUSTAKA

[1] A. Voutama and E. Novalia, "Web-Based Graduation Plaque Information System Design Using UML and Waterfall Model," *Syntax J. Inform.*, vol. 11, no. 01, pp. 36–49, 2022, doi: 10.35706/syji.v11i01.6412.

[2] M. F. Allard and A. Voutama, "Rancang Bangun Sistem Informasi Reservasi Hotel 'Hotel Hebat' Berbasis Website," *J. Inform. dan Tek. Elektro Terap.*, vol. 12, no. 2, 2024, doi: 10.23960/jitet.v12i2.4224.

[3] J. Sains and N. Hidayati, "Aplikasi To-Do List Berbasis Android dengan MIT App Inventor Jurnal Sains dan Teknolog," vol. 01, no. 01, pp. 1–5, 2024.

[4] Z. Afwan, W. J. Kurniawan, and P. P. Putra, "Pembangunan Aplikasi Manajemen Tugas Mahasiswa Berbasis Mobile Android," *J-SAKTI (Jurnal Sains Komput. dan Inform.*, vol. 1, no. 2, p. 203, 2017, doi: 10.30645/j-sakti.v1i2.44.

[5] TRIYAS NIKO SAPUTRA and IMAM HUSNI AL AMIN, "Aplikasi Manajemen Tugas Taskify Untuk Menentukan Prioritas Pekerjaan Freelancer Dengan Ahp Dan Topsis," *J. INSTEK (Informatika Sains dan Teknol.*, vol. 8, no. 2, pp. 292–301, 2023, doi: 10.24252/instek.v8i2.42202.

[6] K. I. R. Costa, "Rancang Bangun Aplikasi Mobile To Do List Sederhana Berbasis Android," *Researchgate.Net*, no. July, pp. 1–6, 2021, [Online]. Available: https://www.researchgate.net/profile/Kevin-Rui-Costa/publication/352961664_RANCANG_BANGUN_APLIKASI_MOBILE_TO_DO_LIST_S

DOI: https://doi.org/10.35457/antivirus.v19i1.4435

EDERHANA_BERBASIS_ANDROID/links/60e10dc3a6fdccb74503830e/RANCANG-BANGUN-APLIKASI-MOBILE-TO-DO-LIST-SEDERHANA-BERBASIS-ANDROID.pdf

[7] I. Ilhamsyah, A. P. Kharisma, and N. Yudistira, "Pengembangan Aplikasi Manajemen Tugas berbasis Android (Studi pada CV. Cheleron Production)," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 5, no. 12, pp. 5221–5227, 2021, [Online]. Available: http://j-ptiik.ub.ac.id

[8] A. L. Permono, D. Ruswanti, and A. Charolina, "Penerapan Mern Stack pada Aplikasi Jadwal Plus sebagai Sistem Manajemen Penjadwalan Berbasis Web," vol. 30, no. 2, pp. 276–286, 2024, doi: 10.36309/goi.v30i2.314.

[9] R. Listiyanto and H. Gunawan, "Perancangan Aplikasi Manajemen Tugas Berbasis Android Menggunakan Metode Agile," *J. Account. Inf. Syst.*, vol. 7, no. 1, pp. 65–72, 2024, doi: 10.32627/aims.v7i1.932.

[10] C. Science and I. Technology, "BOOKSTORE : E-COMMERCE PLATFORM WITH".

[11] V. Virstandi, P. Andanawari, and R. M. Alam, "Perancangan Aplikasi Web E-Commerce Dengan Teknologi Modern Mern Stack," vol. 12, no. 1, pp. 43–54, 2025.

[12] S. Kasus and T. K. Ra, "Aplikasi Bacaan Sholat Untuk Tingkat Paud Berbasis Android Dengan Metode RAD," vol. 13, pp. 2507–2514, 2025.

[13] M. M. Rad, "Aplikasi Manajemen Data Balita Pada Posyandu," vol. 10, no. 1, pp. 71–77, 2024.

[14] P. Ndamunamu, F. Harjadi, and A. C. Talakua, "Aplikasi Berbasis Android Pembelajaran Pengenalan Nama Hewan Menggunakan Metode RAD (Rapid Application Development)," *sudo J. Tek. Inform.*, vol. 2, no. 3, pp. 111–121, 2023, doi: 10.56211/sudo.v2i3.320.

[15] D. Aditya Nugroho and A. Voutama, "Perancangan Aplikasi Berbasis Web Menggunakan Metode SDLC untuk Mengembangkan Sektor Pariwisata Desa Hanau Berak," *Inf. Manag. Educ. Prof.*, vol. 7, no. 2, pp. 154–163, 2023.

[16] I. Djuanda and Dewi, "Perancangan Sistem Informasi Perwira Tugas Belajar Bagi Pegawai Di Kementerian Pertahanan Republik Indonesia," *Semin. Nas. Mhs. Ilmu Komput. dan Apl.*, pp. 655–668, 2020, [Online]. Available: https://repository.upnvj.ac.id/6380/12/AWAL.pdf

[17] D. Hasnan Hariri, H. Hannie, I. Purnamasari, and U. Singaperbangsa Karawang Abstract, "Analisis User Experience pada Website Waste4change Menggunakan Metode Single Ease Question," *J. Ilm. Wahana Pendidik.*, vol. 8, no. 13, pp. 95–108, 2022.

DOI: https://doi.org/10.35457/antivirus.v19i1.4435