JITET (Jurnal Informatika dan Teknik Elektro Terapan)
Vol. 13 No. 3S1, pISSN: 2303-0577 eISSN: 2830-7062 http://dx.doi.org/10.23960/jitet.v13i3S1.7817

# PENGEMBANGAN *BACK-END* APLIKASI MENTALWELL 1.0 BERBASIS *WEBSITE* MENGGUNAKAN *FRAMEWORK* HAPI.JS

**Anindya Kinarya Yang Esa Riyanto[1*], Trisya Septiana[2], Resty Annisa[3]**

[1,2,3] Teknik Informatika, Universitas Lampung, Jl. Prof. Soemantri Brojonegoro, Bandar Lampung 35145

**Keywords:**
Online Counseling; REST API; Hapi.js; JSON Web Token; Agile

**Corespondent Email:**
anin.kinarya@gmail.com

**Abstrak.** Pengembangan aplikasi MentalWell 1.0 dilakukan untuk menghadirkan layanan konseling psikologis berbasis web yang dapat diakses masyarakat. Fokus penelitian ini adalah pada pengembangan sistem *back-end* menggunakan *framework* Hapi.js untuk mengimplementasikan API berarsitektur REST dengan struktur modular dan pengelolaan routing yang terorganisir, sehingga memudahkan proses pengembangan maupun pemeliharaan kode. Keamanan sistem menerapkan JSON Web Token (JWT) sebagai mekanisme autentikasi dan otorisasi berbasis peran. Proses pengembangan dilakukan dengan metode *Agile* melalui enam iterasi, yang terdiri dari pembangunan fitur dasar untuk pasien dan psikolog, penambahan fungsionalitas administrator, pengembangan sistem pemesanan konseling, penerapan mekanisme *soft delete*, serta perbaikan dan optimalisasi fitur berdasarkan umpan balik hasil integrasi dengan *front-end*. Hasil akhir pengembangan menghasilkan 56 modul dengan 36 endpoint untuk peran publik, pasien, psikolog, dan administrator. Seluruh *endpoint* diuji menggunakan Postman dengan total 122 *test case* yang menunjukkan tingkat keberhasilan 100%, dan sistem back-end telah terintegrasi dengan *front-end* untuk mendukung layanan konseling daring secara penuh.



**Abstract.** *The development of MentalWell 1.0 was carried out to provide accessible web-based psychological counseling services for the public. This research focuses on the back-end system development using the Hapi.js framework to implement a REST-architectured API with a modular structure and organized routing management, thereby facilitating both development and maintenance processes. Security is ensured through the application of JSON Web Token (JWT) as the mechanism for role-based authentication and authorization.. The development process was conducted using the Agile method through six iterations, consisting of building basic features for patients and psychologists, adding administrator functionalities, developing the counseling booking system, implementing the soft delete mechanism, as well as improving and optimizing features based on feedback from front-end integration. The final outcome produced 56 modules with 36 endpoints for public, patient, psychologist, and administrator roles. All endpoints were tested using Postman with a total of 122 test cases that achieved a 100% success rate, and the back-end system has been fully integrated with the front-end to support online counseling services.*

## 1. PENDAHULUAN

Kesehatan mental merupakan fondasi penting bagi kesejahteraan individu, yang memungkinkan seseorang berfungsi secara produktif dan mengatasi tantangan hidup. Namun, akses terhadap layanan psikologis profesional masih menghadapi berbagai hambatan signifikan, seperti keterbatasan infrastruktur di daerah terpencil dan adanya

stigma sosial yang menghalangi individu untuk mencari bantuan [1].

Perkembangan teknologi, khususnya aplikasi berbasis web dan *telemedicine*, menawarkan solusi menjanjikan untuk mengatasi hambatan tersebut dengan menyediakan akses layanan yang lebih luas, privat, dan fleksibel [2]. Menjawab kebutuhan ini, MentalWell 1.0 dikembangkan sebagai *platform* konseling psikologis daring yang menghubungkan pengguna dengan psikolog profesional. Namun, versi awal aplikasi ini masih memerlukan sistem *back-end* yang solid untuk mengelola data pengguna, jadwal konseling, dan interaksi secara aman dan terstruktur.

Penelitian ini bertujuan untuk merancang dan membangun sistem *back-end* untuk aplikasi MentalWell 1.0 menggunakan *framework* Hapi.js. Fokus utamanya adalah menciptakan REST API yang modular, mengimplementasikan sistem keamanan menggunakan JSON Web Token (JWT) untuk autentikasi dan otorisasi berbasis peran, merancang *database* yang terstruktur menggunakan PostgreSQL, serta memastikan integrasi dengan aplikasi *front-end*.

## 2. TINJAUAN PUSTAKA

### 2.1 Arsitektur *Back-end* dan REST API

*Back-end* merupakan komponen sisi server dari sebuah aplikasi yang bertanggung jawab atas logika bisnis, interaksi *database*, dan pemrosesan data [3]. *Application Programming Interface* (API) berfungsi sebagai jembatan yang memungkinkan komunikasi antara komponen perangkat lunak yang berbeda [4]. *Representational State Transfer* (REST) adalah gaya arsitektur yang digunakan untuk membangun layanan web yang efisien dan skalabel, dengan prinsip-prinsip seperti *statelessness* dan *uniform interface* menggunakan metode HTTP standar (GET, POST, PUT, DELETE)[5].

### 2.2 *Framework* Hapi.js

Hapi.js adalah *framework open-source* berbasis Node.js yang dikenal dengan pendekatan konfigurasi-sentris (*configuration-centric*)[6]. Berbeda dengan pendekatan *middleware-centric* seperti pada Express.js, Hapi.js mengutamakan pengaturan melalui objek konfigurasi yang membuat struktur kode lebih terorganisir. Fitur bawaan seperti *caching server-side* dan sistem *routing* yang deterministik menjadikan Hapi.js pilihan yang solid untuk aplikasi dengan arsitektur modular dan kebutuhan fitur yang kompleks [7].

### 2.3 Keamanan dengan JSON Web Token (JWT)

JSON Web Token (JWT) adalah standar terbuka (RFC 7519) untuk mentransmisikan klaim secara aman antar pihak dalam format JSON [8]. Sebuah JWT terdiri dari tiga bagian: *header, payload, dan signature.*

*Signature* yang ditandatangani secara digital menggunakan kunci rahasia memastikan integritas data dan memverifikasi identitas pengirim, menjadikannya mekanisme yang andal untuk implementasi autentikasi dan otorisasi dalam sistem [9].

### 2.4 Supabase

Supabase adalah *platform open-source back-end-as-a-service* (BaaS) yang menggunakan PostgreSQL sebagai basis data relasional dan menyediakan layanan seperti PostgreSQL database, *authentication*, storage, *edge function*, dan *realtime*.

a. PostgreSQL

PostgreSQL adalah sistem manajemen basis data relasional *open-source* yang mendukung *query* SQL dan transaksi ACID (*Atomicity, Consistency, Isolation, Durability*). ACID memastikan transaksi dijalankan secara atomik, konsisten, terisolasi, dan tahan lama, mendukung pengelolaan data terstruktur dengan performa tinggi [10].

b. Realtime

Fitur Realtime memungkinkan sinkronisasi data melalui WebSocket dengan memantau perubahan pada tabel *database* [11]. Fitur ini mendukung pembaruan data instan untuk aplikasi yang memerlukan interaksi dinamis.

c. *Storage*

Supabase Storage menyediakan sistem penyimpanan objek berbasis *bucket* untuk mengelola *file*, seperti gambar, video, dan dokumen [12].

### 2.5 Railway

Railway merupakan platform *cloud* berbasis *Platform as a Service* (PaaS) yang dirancang untuk menyederhanakan proses pengembangan

dan *deployment* aplikasi. Railway merupakan *platform deployment* berbasis *cloud* yang mendukung proses otomatisasi *deployment* tanpa memerlukan konfigurasi server secara manual. *Platform* ini menyediakan fitur-fitur seperti integrasi dengan GitHub, *Continuous Deployment*, serta dukungan terhadap berbagai bahasa pemrograman dan layanan basis data seperti PostgreSQL dan MySQL [13].

### 2.6 Postman

Postman adalah *platform* pengembangan dan pengujian API (*Application Programming Interface*) yang dirancang untuk membantu pengembang dalam mengelola, menguji, dan mendokumentasikan API.

Postman mendukung protokol komunikasi seperti HTTP, yang merupakan standar untuk pengujian API berbasis web. Kemampuan mengelola berbagai jenis permintaan, seperti GET, POST, PUT, dan DELETE, membuat Postman menjadi alat fleksibel untuk menguji fungsionalitas API dalam berbagai skenario.

## 3. METODE PENELITIAN

### 3.1 Alat Penelitian

*Tabel 1 Perangkat Keras*

| No. | Alat | Spesifikasi | Fungsi |
|---|---|---|---|
| 1. | Laptop Macbook Air M1 (2020) | macOS Sequoia v.15.5, ARM64, RAM 8GB | Digunakan untuk *coding*, desain, dan pengujian aplikasi |

*Tabel 2 Perangkat Lunak*

| No. | Perangkat Lunak | Fungsi |
|---|---|---|
| 1. | Node.js | *Runtime environment* untuk menjalankan kode JavaScript |
| 2. | Hapi.js | *Framework back-end* untuk membangun REST API |
| 3. | PostgreSQL (via Supabase) | Sistem basis data relasional untuk penyimpanan terstruktur |
| 4. | Visual Studio Code | *Text editor* untuk menulis kode program |
| 5. | Postman | Pengujian dan dokumentasi API |
| 6. | GitHub | *Version control* sekaligus media *deployment* |

*Tabel 3 Paket Dependensi*

| No | Paket | Fungsi Utama |
|---|---|---|
| 1 | @hapi/hapi | Framework inti untuk server dan routing |
| 2 | joi | Validasi data berbasis skema |
| 3 | jsonwebtoken | Autentikasi berbasis token (JWT) |
| 4 | bcryptjs | Hashing dan verifikasi kata sandi |
| 5 | axios | Permintaan HTTP ke API pihak ketiga |

### 3.2 Tahapan Penelitian

Penelitian ini menggunakan metode pengembangan *Agile Agile* memungkinkan proses pengembangan dilakukan secara iteratif dan berkelanjutan, sehingga sistem bisa terus disesuaikan serta ditingkatkan sesuai masukan pengguna. Melalui pendekatan ini, pengembang dapat merespons perubahan kebutuhan dengan lebih cepat sekaligus menjaga agar sistem yang dikembangkan tetap sesuai dan efektif [14]. Setiap iterasi terdiri dari lima tahapan utama: *Plan*, *Design*, *Development*, *Testing*, dan *Deployment*.


*Gambar 1 Siklus Iteratif Agile [15]*

Setiap siklus bertujuan menghasilkan versi perangkat lunak yang dapat digunakan dan dievaluasi. Pada akhir iterasi, proses kembali ke tahap perencanaan untuk memperbaiki atau menambahkan fitur berdasarkan umpan balik pengguna. Siklus berlangsung berulang hingga produk akhir mencapai kualitas dan fungsionalitas yang diharapkan.

## 4. HASIL DAN PEMBAHASAN

### 4.1 *Initial Iteration*

#### 4.1.1 *Plan*

Tahap awal berfokus pada identifikasi kebutuhan sistem melalui studi literatur untuk menentukan pendekatan teknologi yang sesuai, seperti arsitektur REST API, JWT untuk autentikasi, dan Supabase sebagai backend. Hasil analisis kemudian dirumuskan menjadi kebutuhan fungsional dan non-fungsional.

*Tabel 4 Kebutuhan Fungsional*

| Kode Fungsional | Deskripsi |
|---|---|
| KF-01 | Sistem menyediakan fitur layanan konseling dengan profesional. |

| | |
|---|---|
| KF-02 | Sistem menyediakan fitur tes psikologi untuk memahami kondisi mental. |
| KF-03 | Sistem mendukung fitur chat yang digunakan pada sesi konseling. |
| KF-04 | Sistem menyediakan fitur konfirmasi pembayaran pada layanan daftar konseling. |

*Tabel 5 Kebutuhan Non-Fungsional*

| Kode Non-Fungsional | Deskripsi |
|---|---|
| KNF-01 | Sistem menggunakan JSON Web Token untuk autentikasi dan otorisasi pengguna. |
| KNF-02 | Semua pertukaran data dilakukan dalam format JSON. |
| KNF-03 | Database didesain relasional untuk menjaga konsistensi data. |
| KNF-04 | Sistem dirancang dengan struktur modular. |
| KNF-05 | Sistem menjalankan tugas terjadwal *(scheduled task)* untuk menjalankan pembaruan status sesi konseling secara berkala tanpa input manual. |
| KNF-06 | Sistem di*deploy* ke *platform cloud* dan dapat diakses oleh aplikasi *front-end* melalui internet |
| KNF-07 | Setiap *endpoint back-end* didokumentasikan menggunakan Postman untuk memudahkan proses integrasi dan pengujian. |

### 4.1.2 *Design*

Tahapan ini meliputi perancangan untuk memberikan gambaran menyeluruh mengenai alur kerja, struktur data, serta interaksi komponen di dalam sistem sebelum tahap implementasi dilakukan.

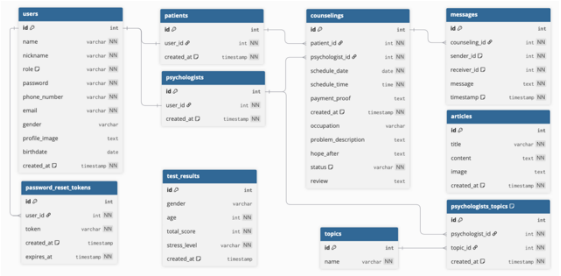
a. *Use Case Diagram*

*Use case diagram* menunjukkan hubungan antara aktor dengan fungsionalitas yang tersedia pada aplikasi MentalWell 1.0. Diagram tersebut ditampilkan pada Gambar 2.


*Gambar 2 Use Case Diagram MentalWell 1.0*

b. *Entity Relationship Diagram* (ERD)

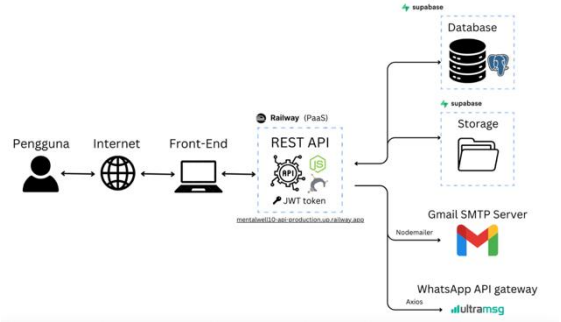
ERD digunakan untuk menggambarkan struktur basis data beserta relasi antar tabel pada aplikasi MentalWell 1.0.




*Gambar 3 ERD MentalWell 1.0*

c. Perencanaan Arsitektur *Back-End*

Arsitektur *back-end* MentalWell 1.0 dirancang untuk memberikan gambaran alur kerja sistem serta bagaimana setiap komponen saling terhubung. Arsitektur *back-end* tersebut ditampilkan pada Gambar 4.




*Gambar 4 Arsitektur Back-End MentalWell 1.0*

### 4.1.3 *Development*

Tahap *development* menghasilkan implementasi fitur inti berupa autentikasi (*registrasi*, *login*, *reset password*), pengelolaan profil, serta akses artikel. Seluruh layanan dibangun menggunakan arsitektur REST API dan dapat diintegrasikan dengan aplikasi *client*. Daftar *endpoint* yang diimplementasikan ditunjukkan pada tabel-tabel dibawah ini.

*Tabel 6 Endpoint Publik*

| No | *Endpoint* | *Method* | Deskripsi |
|---|---|---|---|
| 1. | */register* | *POST* | *Registrasi* pengguna baru |
| 2. | */login* | *POST* | Autentikasi pengguna dengan JWT |
| 3. | */forgot-password* | *POST* | Permintaan *reset password* |
| 4. | */reset-password* | *POST* | Mengatur ulang *password* pengguna |

| | | | |
|---|---|---|---|
| 5. | */articles* | *GET* | Mengambil daftar artikel |
| 6. | */articles/:id* | *GET* | Mengambil detail artikel berdasarkan ID |

*Tabel 7 Endpoint Pasien*

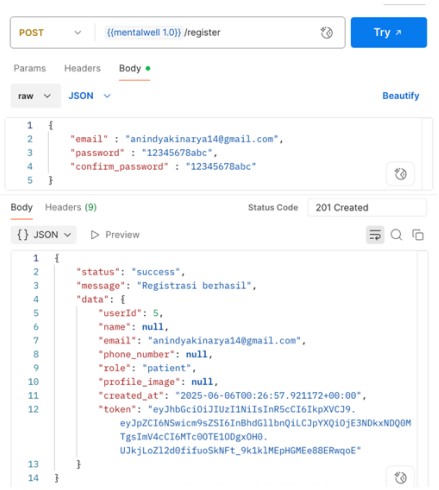
| No | ***Endpoint*** | ***Method*** | **Deskripsi** |
|---|---|---|---|
| 1. | */profile* | *GET* | Mengambil data profil pengguna |
| 2. | */profile* | *PUT* | Memperbarui data profil pengguna |
| 3. | */my-data* | *GET* | Mengambil data akun pengguna yang sedang login |
| 4. | */psychologists* | *GET* | Mengambil daftar psikolog |
| 5. | */psychologists/:id* | *GET* | Mengambil detail psikolog berdasarkan ID |
| 6. | */counseling/:id* | *POST* | Membuat permintaan konseling dengan psikolog tertentu |
| 7. | */counselings* | *GET* | Mengambil daftar sesi konseling |
| 8. | */counselings/:id* | *GET* | Mengambil detail sesi konseling berdasarkan ID |
| 9. | */counselings/:id /review* | *POST* | Memberikan ulasan terhadap sesi konseling |

*Tabel 8 Endpoint Psikolog*

| No | ***Endpoint*** | ***Method*** | **Deskripsi** |
|---|---|---|---|
| 1. | */psychologist /availability* | *PUT* | Memperbarui ketersediaan psikolog |
| 2. | */psychologist /counselings* | *GET* | Mengambil daftar sesi konseling milik psikolog |

### 4.1.4 *Testing*

Pengujian fungsional dilakukan secara manual menggunakan Postman untuk memastikan respons API sesuai dengan skenario *positive case* maupun *negative case*. Sebagai contoh, pada endpoint *POST /register*, sistem berhasil mengembalikan status 201 *Created* beserta token JWT saat diberikan input valid.




*Gambar 5 Contoh Pengujian Endpoint /register menggunakan Postman*

Secara keseluruhan, pengujian pada tahap awal mencakup 17 *endpoint* dengan total 40 *test case*, melibatkan peran publik, pasien, dan psikolog. Hasil pengujian menunjukkan seluruh skenario berjalan sesuai harapan dengan tingkat keberhasilan 100%.

## 4.2 Iterasi 1

### 4.2.1 *Plan*

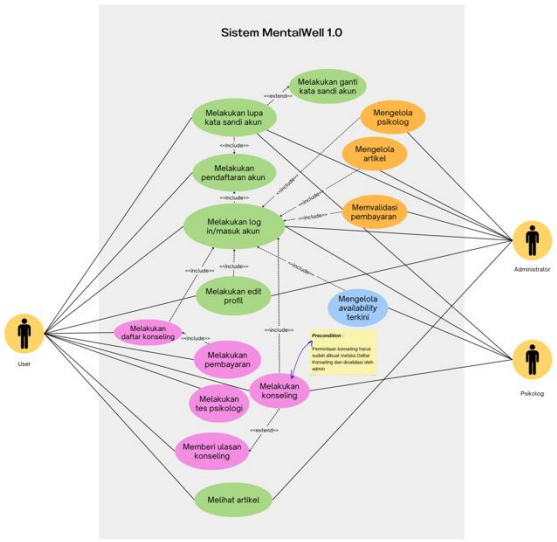
Pada Iterasi 1, perencanaan difokuskan pada penambahan peran administrator untuk mengelola data psikolog, artikel, serta memvalidasi pembayaran, tanpa mengubah fungsionalitas yang sudah ada bagi pasien dan psikolog.

### 4.2.2 *Design*

Perancangan pada iterasi ini melibatkan pembaruan pada beberapa diagram.

a. *Use Case Diagram*

*Use Case Diagram* diperbarui dengan aktor Administrator yang memiliki tiga *use case*: mengelola psikolog, mengelola artikel, dan memvalidasi pembayaran.




*Gambar 6 Use Case pada Iterasi 1*

b. *Entity Relationship Diagram* (ERD)

*Gambar 7 ERD pada Iterasi 1*

### 4.2.3 *Development*

Pada tahap ini dikembangkan fitur administrator berupa *endpoint* untuk mengelola artikel dan memantau konseling. Daftar *endpoint* ditunjukkan pada Tabel 9.

*Tabel 9 Endpoint Administrator*

| No | *Endpoint* | Method | Deskripsi |
|---|---|---|---|
| 1. | */article* | POST | Menambahkan artikel baru |
| 2. | */article/:id* | PUT | Memperbarui artikel berdasarkan ID |
| 3. | */article/:id* | DELETE | Menghapus artikel berdasarkan ID |
| 4. | */counselings* | GET | Mengambil daftar sesi konseling |
| 5. | */counselings/:id* | GET | Mengambil detail sesi konseling berdasarkan ID |

### 4.2.4 *Testing*

Pengujian Iterasi 1 difokuskan pada manajemen artikel dan konseling oleh administrator. Sebanyak 5 *endpoint* diuji dengan total 14 *test case* mencakup input valid, tidak valid, dan validasi akses. Seluruh pengujian berhasil dengan tingkat keberhasilan 100%.

## 4.3 Iterasi 2

### 4.3.1 *Plan*

Iterasi 2 difokuskan pada perbaikan mekanisme pemesanan konseling agar lebih terstruktur. Sistem dikembangkan untuk mengelola ketersediaan psikolog baik mingguan, per tanggal, maupun *realtime* sesuai ketersediaan psikolog saat ini, sehingga pasien hanya dapat memilih jadwal yang benar-benar tersedia.

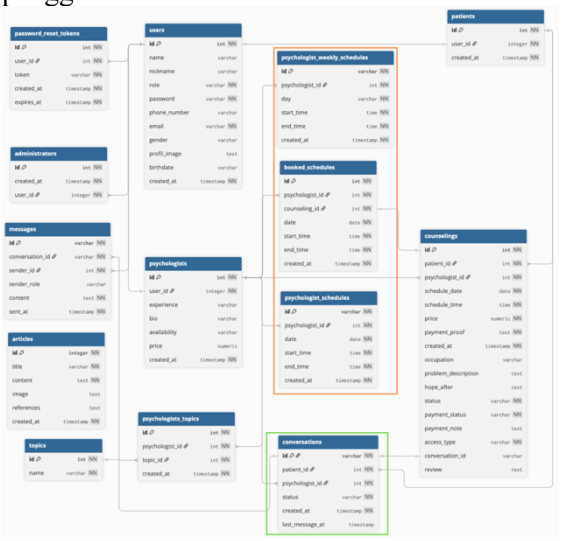
### 4.3.2 *Design*

a. *Entity Relationship Diagram* (ERD)

*Database* diperbarui secara signifikan untuk mendukung sistem penjadwalan yang baru. Perubahan ini meliputi penambahan tiga tabel baru:

- *psychologist_weekly_schedules*: Untuk menyimpan jadwal ketersediaan rutin mingguan psikolog.
- *psychologist_schedules*: Untuk menyimpan jadwal ketersediaan pada tanggal tertentu.
- *booked_schedules*: Untuk mencatat jadwal yang sudah dipesan oleh pasien guna mencegah *double-booking*.

Selain itu, ditambahkan tabel *conversations* untuk mengelola riwayat percakapan antar pengguna secara lebih terstruktur.



*Gambar 8 ERD pada Iterasi 2*

### 4.3.3 *Development*

Pada tahap pengembangan, ditambahkan dua jenis layanan pemesanan konseling: konseling terjadwal dan *chat now* (*realtime*). Berikut adalah daftar *endpoint* yang diimplementasikan pada iterasi ini.

*Tabel 10 Endpoint pada Iterasi 2 (Pasien)*

| No | *Endpoint* | *Method* | Deskripsi |
|---|---|---|---|
| 1. | */psychologists /:id/schedules* | *GET* | Mengambil daftar jadwal psikolog |
| 2. | */psychologists/:id /schedules /availability* | *GET* | Mengecek ketersediaan jadwal psikolog |
| 3. | */counseling/:id* | *POST* | Membuat permintaan konseling terjadwal dengan psikolog |
| 4. | */realtime /counseling/:id* | *POST* | Membuat permintaan konseling *realtime* dengan psikolog |
| 5. | */counselings/:id* | *GET* | Mengambil detail sesi konseling berdasarkan ID |

*Tabel 11 Endpoint pada Iterasi 2 (Psikolog)*

| No | *Endpoint* | *Method* | Deskripsi |
|---|---|---|---|
| 1 | */psychologist /counselings/:id* | *GET* | Mengambil detail sesi konseling milik psikolog |
| 2 | */psychologist /counselings /:id/status* | *PUT* | Memperbarui status sesi konseling oleh psikolog |

*Tabel 12 Endpoint pada Iterasi 2 (Administrator)*

| No | *Endpoint* | *Method* | Deskripsi |
|---|---|---|---|
| 1 | */administrator /counselings/:id* | *PUT* | Memperbarui data atau status sesi konseling |
| 2 | */administrator /psychologist* | *POST* | Menambahkan data psikolog baru |
| 3 | */administrator /psychologists/:id* | *GET* | Mengambil detail data psikolog berdasarkan ID |
| 4 | */administrator /psychologists/:id* | *PUT* | Memperbarui data psikolog berdasarkan ID |

Selain itu, diimplementasikan juga tugas terjadwal (*scheduled task*) yang berjalan setiap menit untuk memperbarui status sesi konseling secara otomatis.

### 4.3.4 *Testing*

Pada Iterasi 2 diuji 11 *endpoint* yang melibatkan pasien, psikolog, dan administrator. Pengujian mencakup pemesanan konseling, pengelolaan jadwal, data psikolog, serta validasi pembayaran dengan total 42 *test case*. Seluruh pengujian berhasil dijalankan sesuai ekspektasi dengan tingkat keberhasilan 100%.

## 4.4 Iterasi 3

### 4.4.1 *Plan*

Iterasi 3 difokuskan pada optimalisasi sistem dengan penerapan *soft delete* pada data psikolog. Mekanisme ini menggunakan kolom *is_active* pada tabel *users* dan database *view* untuk hanya menampilkan psikolog yang aktif, sehingga integritas data relasional tetap terjaga.

### 4.4.2 *Development*

a. Implementasi *Database View*

Untuk menyederhanakan *query* yang kompleks dan berulang, sebuah *database view* bernama *psychologists_view* dibuat. *View* ini berfungsi sebagai tabel virtual yang menggabungkan data dari beberapa tabel inti seperti *users, psychologists, topics*, dan jadwal. Kegunaan utamanya adalah *view* ini secara otomatis telah menyaring data untuk hanya menampilkan psikolog dengan status *is_active = true*, sehingga memastikan setiap *endpoint* yang menggunakannya hanya akan menyajikan data yang relevan.

b. Implementasi pada *Endpoint* API

*View* ini menjadi dasar bagi pengembangan empat *endpoint* utama:

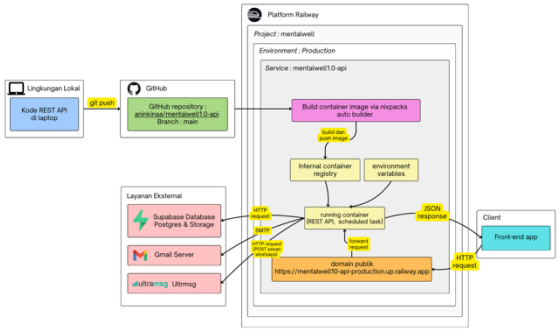
- *GET /psychologists/list*: Menyediakan daftar semua psikolog aktif untuk pasien, menggantikan endpoint lama yang tidak efisien.
- *GET /psychologists/search*: Memungkinkan pasien mencari psikolog berdasarkan nama dan/atau topik keahlian dengan logika penyaringan dua tahap (pencarian '*AND'*, jika gagal dilanjutkan '*OR'*).
- *GET /administrator/psychologists*: Menyediakan daftar psikolog aktif dengan kolom data yang spesifik untuk keperluan administratif.
- *DELETE /administrator/psychologists/:id:* Mengimplementasikan mekanisme *soft delete*. Proses ini tidak menghapus data secara permanen, melainkan mengubah status *is_active* psikolog menjadi *false* di tabel *users* dan membersihkan data relasional terkait (seperti jadwal dan topik).

#### 4.4.3 *Testing*

Pada iterasi 3, dilakukan pengujian fungsional pada 4 *endpoint* untuk dua peran, yaitu pasien dan administrator. Hasil pengujian menunjukkan seluruh fungsi berjalan sesuai ekspektasi, dengan tingkat keberhasilan 100%.

#### 4.4.4 *Deployment*

Pada tahap ini aplikasi *back-end* REST API di-*deploy* ke *platform* Railway, membuatnya dapat diakses secara publik melalui URL yang disediakan.




*Gambar 9 Arsitektur Deployment Back-End MentalWell 1.0*

### 4.5 Iterasi 4

#### 4.5.1 *Plan*

Iterasi 4 difokuskan pada perbaikan *bug* dan penyempurnaan fungsionalitas berdasarkan umpan balik dari tim *front-end* setelah proses integrasi awal. Perencanaan pengembangan mencakup beberapa poin utama: penambahan fitur kategori artikel, perbaikan alur autentikasi dan otorisasi pada beberapa *endpoint*, serta perbaikan bug pada fitur lupa kata sandi dan penyimpanan pesan *chat*.

#### 4.5.2 *Development*

Beberapa pengembangan dan perbaikan di iterasi ini meliputi:

- Fitur Kategori Artikel: Penambahan tabel categories dan articles_categories untuk relasi many-to-many, serta pembaruan endpoint CRUD artikel agar mendukung pengelolaan kategori.
- Perbaikan Bug Pesan: Koreksi trigger database update_last_message_at sehingga penyimpanan pesan di tabel messages berjalan normal.
- Perbaikan Alur Lupa Sandi: Tautan reset password diperbarui agar mengarah ke *route front-end* yang benar (/ubah-sandi).
- Penyesuaian *Endpoint* dan Hak Akses: *GET /psychologists/list* kini publik, *PUT /profile* diperluas untuk administrator, dan respons login disempurnakan menyertakan nama serta peran pengguna.

#### 4.5.3 *Testing*

Pada iterasi 4, pengujian difokuskan untuk memastikan perbaikan fungsional dari umpan balik sebelumnya. Sebanyak 9 *endpoint* diuji melalui 14 *test case*, meliputi fitur artikel, *login*, profil administrator, dan lupa sandi. Hasil pengujian seluruhnya sesuai ekspektasi, dengan tingkat keberhasilan 100%.

### 4.6 Iterasi 5

#### 4.6.1 *Plan*

Iterasi 5 berfokus pada peningkatan performa dan keandalan sistem, terutama pada fitur pengiriman email. Masalah keterlambatan dengan SMTP Gmail diatasi melalui migrasi ke *platform* Brevo, yang menyediakan API email transaksional lebih cepat.

#### 4.6.2 *Development*

Pada iterasi ini, proses pengembangan utama adalah integrasi dengan Brevo API. Nodemailer dan SMTP Gmail digantikan dengan fungsi baru menggunakan Axios untuk mengirim permintaan HTTP POST ke Brevo. API key disimpan sebagai environment variable, dan fungsi ini diintegrasikan ke fitur notifikasi seperti lupa sandi dan konfirmasi pembayaran.

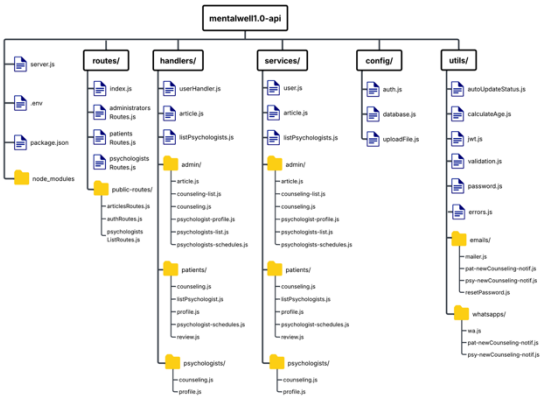
#### 4.6.3 *Testing*

Pada iterasi ini, pengujian difokuskan pada validasi migrasi layanan email melalui endpoint *POST /forgot-password* dan *PUT /administrator/counseling/:id*. Hasil menunjukkan email diterima dalam maksimal 2 menit, lebih cepat dari SMTP Gmail sebelumnya, dengan tingkat keberhasilan 100%.

### 4.7 Rekapitulasi

Hasil pengembangan *back-end* aplikasi MentalWell 1.0 menghasilkan 56 modul dengan total 36 *endpoint* REST API yang terbagi ke dalam peran publik, pasien, psikolog, dan administrator. Gambaran umum struktur direktori ditunjukkan pada Gambar 10,

sedangkan rincian lengkap setiap *endpoint* ditampilkan pada Tabel 13–Tabel 16.




*Gambar 10 Struktir Direktori Aplikasi Back-End MentalWell 1.0*

*Tabel 13 Endpoint Publik*

| No. | *Method* | *Endpoint* | Deskripsi |
|---|---|---|---|
| 1. | *POST* | */register* | Registrasi akun pasien |
| 2. | *POST* | */login* | Login & dapatkan JWT |
| 3. | *POST* | */forgot-password* | Permintaan reset password |
| 4. | *POST* | */reset-password* | Reset password dengan token |
| 5. | *GET* | */articles* | Melihat daftar artikel |
| 6. | *GET* | */articles/:id* | Melihat detail artikel |
| 7. | *GET* | */psychologists /list* | Melihat daftar psikolog |
| 8. | *GET* | */psychologists /search* | Cari psikolog berdasarkan nama/topik |

*Tabel 14 Endpoint Pasien*

| No. | *Method* | *Endpoint* | Deskripsi |
|---|---|---|---|
| 1. | *GET* | */profile* | Melihat profil pasien |
| 2. | *PUT* | */profile* | Edit profil pasien |
| 3. | *GET* | */my-data* | Autofill data pasien untuk konseling |
| 4. | *GET* | */psychologists* | Lihat daftar psikolog |
| 5. | *GET* | */psychologists/ :id* | Lihat detail psikolog |
| 6. | *POST* | */counseling/:id* | Pesan konseling terjadwal |
| 7. | *GET* | */counselings* | Lihat daftar konseling pasien |
| 8. | *GET* | */counselings /:id* | Lihat detail konseling pasien |
| 9. | *POST* | */counseling/:id /review* | Tambah ulasan konseling |
| 10. | *GET* | */psychologists /:id/schedules* | Lihat jadwal psikolog |
| 11. | *GET* | */psychologists /:id/schedules /availaibility* | Lihat ketersediaan jadwal psikolog |
| 12. | *POST* | */realtime /counseling/:id* | Pesan konseling realtime |
| 13. | *GET* | */psychologists /list* | Lihat daftar psikolog (versi list view) |

*Tabel 15 Endpoint Psikolog*

| No. | *Method* | *Endpoint* | Deskripsi |
|---|---|---|---|
| 1. | *GET* | */psychologist /profile* | Lihat profil psikolog |
| 2. | *PUT* | */psychologist /availability* | Edit ketersediaan psikolog |
| 3. | *GET* | */psychologist /counselings* | Lihat daftar konseling psikolog |
| 4. | *GET* | */psychologist /counselings /:id* | Lihat detail konseling psikolog |
| 5. | *PUT* | */psychologist /counselings /:id /status* | Update status konseling |

*Tabel 16 Endpoint Administrator*

| No. | *Method* | *Endpoint* | Deskripsi |
|---|---|---|---|
| 1. | *POST* | */article* | Buat artikel |
| 2. | *PUT* | */article/:id* | Edit artikel |
| 3. | *DELETE* | */article/:id* | Hapus artikel |
| 4. | *GET* | */administrator /counselings* | Lihat semua data konseling |
| 5. | *GET* | */administrator /counselings/:id* | Lihat detail konseling |
| 6. | *PUT* | */administrator /counselings/:id* | Update status pembayaran |
| 7. | *POST* | */administrator /psychologist* | Tambah data psikolog |
| 8. | *GET* | */administrator /psychologists* | Lihat daftar psikolog |
| 9. | *GET* | */administrator /psychologists/:id* | Lihat detail psikolog |

| | | | |
|---|---|---|---|
| 10. | *PUT* | */administrator /psychologists/:id* | Edit detail psikolog |
| 11. | *DELETE* | */administrator /psychologists/:id* | Hapus psikolog |

Pengujian fungsional dilakukan secara iteratif sepanjang proses pengembangan untuk memastikan setiap endpoint bekerja sesuai harapan. Total keseluruhan pengujian mencakup 122 test case yang tersebar di beberapa iterasi:

- *Initial Iteration*: 40 *test case*
- Iterasi 1: 14 *test case*
- Iterasi 2: 42 *test case*
- Iterasi 3: 11 *test case*
- Iterasi 4: 14 *test case*
- Iterasi 5: 2 *test case*

Seluruh 122 *test case* berhasil dijalankan dengan hasil 100% sukses, sehingga dapat disimpulkan bahwa seluruh endpoint yang telah diimplementasikan berfungsi sesuai spesifikasi.

## 5. KESIMPULAN

Kesimpulan yang didapatkan pada penelitian ini adalah sebagai berikut.

a. *Framework* Hapi.js berhasil digunakan untuk membangun REST API yang modular dan terintegrasi pada aplikasi MentalWell 1.0. dengan total 56 modul yang dikembangkan.
b. Sistem mendukung seluruh fitur autentikasi, manajemen profil, pemesanan konseling, artikel, pembayaran, dan notifikasi. Pengujian pada 36 endpoint dengan 122 *test case* menunjukkan tingkat keberhasilan 100%.
c. Kekurangan sistem adalah masih bergantung pada layanan gratis Supabase dan Railway yang memiliki keterbatasan kapasitas, serta metode pembayaran yang masih manual. Pengembangan selanjutnya dapat difokuskan pada optimasi performa dan integrasi *payment gateway* agar lebih praktis dan otomatis.

## UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada pihak-pihak terkait yang telah memberi dukungan terhadap penelitian ini.

## DAFTAR PUSTAKA

[1] C. M. Green, C. A. Hostutler, K. L. Lovero, J. A. Mautone, and R. Platt, “Editorial: Pediatric integrated care: from vision to practice,” *Frontiers in Psychiatry*, vol. 15, June 2024.
[2] N. Lisnarini, J. R. Suminar, and Y. Setianti, “Keunggulan dan Hambatan Komunikasi dalam Layanan Kesehatan Mental pada Aplikasi Telemedicine Halodoc,” *Psikobuletin:Buletin Ilmiah Psikologi*, vol. 4, no. 3, Sept. 2023.
[3] I. Kurniawan and F. Rozi, “REST API Menggunakan NodeJS pada Aplikasi Transaksi Jasa Elektronik Berbasis Android,” vol. 1, no. 4, 2020.
[4] M. F. Albaihaqi, “Rancang Bangun RESTful API dan Website Admin untuk Aplikasi Precision Agriculture,” 2022.
[5] F. Doglio, *REST API Development with Node.js: Manage and Understand the Full Capabilities of Successful REST Development*. Apress, 2018.
[6] M. Harrison, *Hapi.js in Action*. Manning Publications Co, 2017.
[7] C. J. Ihrig, *Full Stack JavaScript Development with MEAN: MongoDB, Express, AngularJS, and Node. JS*, 1st ed. Victoria: SitePoint Pty, Limited, 2015.
[8] R. Kumar, *Ultimate Node. js for Cross-Platform App Development: Learn to Build Robust, Scalable, and Performant Server-Side JavaScript Applications with Node. js (English Edition)*, 1st ed. Delhi: Orange Education PVT Ltd, 2024.
[9] M. Fahreza, “Penerapan REST API Menggunakan JSON Web Token,” *Fakultas Sains Teknologi UIN Syarif Hidayatullah Jakarta*, 2023.
[10] “About,” PostgreSQL. Accessed: Mar. 20, 2025. [Online]. Available: https://www.postgresql.org/about/
[11] “Realtime,” Supabase. Accessed: Mar. 18, 2025. [Online]. Available: https://supabase.com/realtime
[12] “Storage Quickstart,” Supabase Docs. Accessed: Mar. 18, 2025. [Online]. Available: https://supabase.com/docs/guides/realtime
[13] Railway, “About Railway,” Railway Documentation. Accessed: Mar. 17, 2025. [Online]. Available: https://docs.railway.app/overview/about-railway
[14] R. Pratama, “Perancangan Dan Implementasi Sistem Manajemen Cuti Pegawai Berbasis Web Menggunakan Pendekatan Agile,” *JITET*, vol. 12, no. 3, Aug. 2024, doi: 10.23960/jitet.v12i3.4527.
[15] R. Rahman, Niswa Ayu Lestari, and H. Hastuti, “Implementasi Metode Agile dan Framework

Laravel Pada Pengembangan Sistem Informasi Bimbingan Skripsi Berbasis Web," *teknimedia*, vol. 5, no. 2, Dec. 2024.