DOKUMEN TEKNIS

Automated Cyber Security Maturity Assessment (AMATI)

Daftar Isi

Daftar Gambar

Daftar Tabel

A. Pendahuluan

1. Deskripsi Proyek

Automated Cyber Security Maturity Assessment (AMATI) adalah proyek dengan tujuan mengembangkan sebuah aplikasi yang memberikan solusi keamanan siber secara efektif dan efisien dengan mengotomatisasi proses audit keamanan manual menjadi berbasis website. AMATI merupakan aplikasi yang memudahkan pengguna dalam mengakses dan melakukan asesmen audit keamanan. Selain itu, fitur otomatisasi ini memanfaatkan teknologi AI yang mampu meminimalkan keterlibatan manusia untuk menganalisis data secara real-time.

2. Tujuan Proyek

Tujuan utama dari pengembangan sistem *Automated Cyber Security Maturity Assessment* (AMATI) adalah untuk menyediakan solusi keamanan siber yang efektif dan efisien, dengan memanfaatkan kecerdasan buatan untuk mengotomatisasi proses audit keamanan. Dengan fitur otomatisasi ini, AMATI bertujuan untuk meminimalkan keterlibatan manusia dalam proses audit, mengurangi potensi kesalahan manusia, dan meningkatkan efisiensi operasional.

AMATI dirancang untuk mendeteksi celah keamanan secara cepat dengan memanfaatkan teknologi AI yang mampu menganalisis data secara real-time, memungkinkan respons yang cepat dan penyediaan rekomendasi mitigasi secara otomatis. Hal ini penting untuk menjaga keamanan sistem dari ancaman yang terus berkembang, serta memastikan bahwa organisasi dapat memenuhi standar kepatuhan keamanan baik di tingkat nasional maupun internasional. Dengan demikian, AMATI bertujuan untuk memberikan perlindungan yang komprehensif terhadap risiko kebocoran data dan serangan siber, sekaligus mendukung organisasi dalam mencapai tingkat kesiapan keamanan siber yang lebih tinggi.

Selain itu, AMATI berambisi untuk mencapai Tingkat Kesiapan Teknologi (TKT) 6 pada tahun 2024, dan terus berkembang ke TKT 7 pada tahun 2025.

Dengan pencapaian ini, AMATI akan dilengkapi dengan teknologi AI dan protokol keamanan terbaru, sehingga mampu memberikan perlindungan yang lebih baik dan memastikan bahwa setiap aspek keamanan siber dapat diatasi secara proaktif dan adaptif.

3. Lingkup Proyek

a. Lingkup Frontend

Frontend bertindak sebagai antarmuka interaktif untuk pengguna umum (user/responden) dan admin. Fokusnya adalah memastikan pengalaman pengguna yang ramah, cepat, dan efisien.

1. Fitur Frontend untuk User/Responden:
    - ➢ Daftar Akun:
        - Form pendaftaran untuk memasukkan biodata sesuai data perusahaan.
        - Validasi input untuk memastikan data yang dimasukkan benar.
    - ➢ Sign In:
        - Validasi username/email dan password.
        - Penyimpanan token autentikasi di database.
    - ➢ Lupa Kata Sandi:
        - Form untuk memasukkan email terdaftar.
        - Tautan reset kata sandi dikirim ke email pengguna.
    - ➢ Permohonan Asesmen:
        - Form untuk mengajukan permohonan asesmen.
        - Status permohonan (Menunggu Konfirmasi, Diterima, Ditolak).
    - ➢ Halaman Asesmen:
        - Tampilan daftar pertanyaan sesuai aspek dan sub-aspek.
        - Dukungan unggah file opsional untuk setiap pertanyaan. Dan juga validasi untuk setiap file yang diunggah hanya bisa menggunakan format

JPG, PNG, PDF. Dan ukuran maksimal file yang bisa diupload adalah 64MB.

- Navigasi paginasi responsif (10 pertanyaan per halaman).
- Validasi jumlah soal yang sudah terjawab.

➢ Hasil Asesmen:

- Tabel skor hasil asesmen.
- Grafik visualisasi hasil.

2. Fitur Frontend untuk Admin:

➢ Manajemen Pengguna:

- Daftar seluruh pengguna (user dan superadmin).
- Tambah, edit, dan hapus pengguna.

➢ Manajemen Aspek:

- Tambah, edit, dan hapus aspek dan sub-aspek.
- Validasi sub aspek yang sudah dibuat tidak boleh sama untuk setiap aspek.

➢ Manajemen Pertanyaan:

- Tambah, edit, dan hapus pertanyaan.
- Dukungan untuk pertanyaan dengan file upload opsional.

➢ Manajemen Permohonan Asesmen:

- Validasi data permohonan asesmen pengguna.
- Opsi untuk menerima atau menolak permohonan.

➢ Manajemen Hasil Asesmen:

- Validasi jawaban asesmen.
- Perubahan jawaban jika ditemukan ketidaksesuaian.
- Tabel dan grafik hasil asesmen.
- Opsi untuk mengekspor hasil asesmen ke PDF.

b. Lingkup Backend

Backend menangani logika bisnis, autentikasi, penyimpanan data, dan komunikasi antara frontend dan database.

1. Fitur Backend untuk User/Responden:

- ➢ Autentikasi dan Otorisasi:
    - Pendaftaran akun baru.
    - Validasi login dengan username/email dan password.
    - Reset password melalui token yang dikirim ke email.
- ➢ Permohonan Asesmen:
    - Endpoint untuk mengirim permohonan asesmen.
    - Validasi data permohonan.
    - Penyimpanan status permohonan.
- ➢ Halaman Asesmen:
    - Penyediaan data aspek, sub-aspek, dan pertanyaan.
    - Penyimpanan jawaban asesmen.
    - Penyimpanan file yang diunggah (jika ada).
- ➢ Hasil Asesmen:
    - Penyediaan data hasil asesmen dalam format tabel dan grafik.
    - Ekspor data hasil asesmen menjadi PDF.

4. Teknologi yang digunakan

   Pengembangan dan implementasi sistem AMATI memerlukan berbagai perangkat keras dan perangkat lunak yang akan mendukung kelancaran proses pengembangan, pengujian, dan operasional. Berikut adalah rincian perangkat yang dibutuhkan:

   a. Kebutuhan Perangkat Keras:
      - Komputer Pengembang:
        - Spesifikasi: Laptop atau desktop dengan spesifikasi yang memadai, seperti prosesor modern, RAM minimal 8GB, dan penyimpanan yang cukup untuk menginstal berbagai perangkat lunak pengembangan.
        - Fungsi: Digunakan oleh tim pengembang untuk mengembangkan, menguji, dan memelihara aplikasi.

- Perangkat Pengujian:
    - Spesifikasi: Perangkat mobile dan desktop dengan berbagai sistem operasi (Windows, Linux, Android) untuk memastikan aplikasi dapat diakses dan berfungsi dengan baik di berbagai platform.
    - Fungsi: Digunakan untuk pengujian fungsionalitas, responsivitas, dan performa aplikasi di berbagai lingkungan.

b. Kebutuhan Perangkat Lunak

➢ Frontend:

- React: Pustaka JavaScript untuk membangun antarmuka pengguna yang responsif dan dinamis.
- TanStack Router: Router modern untuk aplikasi React yang menyediakan manajemen rute, status, dan data yang lebih kuat dan fleksibel. TanStack Router memungkinkan penanganan rute yang efisien, memisahkan logika rute dari UI, dan memungkinkan pengelolaan status aplikasi yang lebih baik.
- Tailwind CSS: Framework CSS utilitas yang memungkinkan pengembangan antarmuka pengguna yang responsif dan dapat disesuaikan. Tailwind CSS menyediakan kelas-kelas siap pakai untuk styling elemen, sehingga meningkatkan produktivitas dalam merancang antarmuka.
- ShadCN: Komponen UI untuk React yang memanfaatkan Radix UI dan Tailwind CSS untuk membuat antarmuka pengguna yang mudah disesuaikan dan memiliki pengalaman pengguna yang sangat baik. ShadCN menyediakan komponen siap pakai yang fleksibel dan dapat dikustomisasi, memungkinkan pengembang untuk membangun aplikasi dengan antarmuka yang konsisten dan efisien.

- Vite: Alat build dan pengembangan frontend yang mempercepat waktu build dan memberikan pengalaman pengembangan yang lebih efisien dengan hot module replacement dan ES modules.
- Alat Pengembangan: Text editor atau IDE (Integrated Development Environment) seperti Visual Studio Code untuk pengembangan kode.

➢ Backend:

- Hono: Framework yang digunakan untuk membangun API dan logika backend.
- Node.js: Lingkungan runtime yang diperlukan untuk menjalankan aplikasi Hono.
- Drizzle ORM: ORM (Object-Relational Mapping) yang memungkinkan interaksi dengan database menggunakan pola objek. Drizzle ORM memudahkan pengelolaan query database, menyediakan fitur seperti migrasi schema, dan meningkatkan produktivitas pengembangan backend dengan API yang sederhana dan konsisten.
- Middleware: Beberapa middleware yang diperlukan untuk menangani autentikasi, logging, dan pengaturan CORS

➢ Database:

- PostgreSQL: Sistem manajemen basis data relasional yang akan menyimpan data pengguna, hasil asesmen, dan informasi lainnya.
- Alat Administrasi Database: Tools seperti pgAdmin atau DBeaver untuk mengelola database PostgreSQL dengan lebih mudah.

➢ Kecerdasan Buatan (AI):

- LLaMA (Large Language Model Meta AI): model bahasa yang dikembangkan oleh Meta (Facebook) untuk tugas-tugas pemrosesan bahasa alami.

➢ Sistem Manajemen Versi:

- Git: Alat untuk mengelola versi kode dan kolaborasi antar pengembang.
- Platform Kolaborasi: GitHub atau GitLab untuk hosting repositori dan manajemen proyek

➢ Alat Pengujian:

- Postman: Alat untuk menguji API yang dikembangkan di backend

➢ Perangkat Lainnya:

1. Alat Kolaborasi Tim:
   - Platform Komunikasi: Slack, Microsoft Teams, atau alat serupa untuk memfasilitasi komunikasi antar anggota tim selama pengembangan.
   - Alat Manajemen Proyek: Trello, JIRA, atau Asana untuk mengelola tugas dan waktu proyek.
2. Dokumentasi:
   - Wiki atau Google Docs: Untuk mendokumentasikan proses pengembangan, spesifikasi teknis, dan panduan pengguna.

B. Prasyarat

1. Frontend

a. Paket manager:

- npm (Node Package Manager) adalah package manager bawaan untuk Node.js untuk mengelola dependensi (library, modul) dalam proyek JavaScript.
- pnpm adalah alternatif package manager yang dirancang lebih cepat dan efisien dalam penggunaan ruang penyimpanan. Versi yang direkomendasikan adalah versi 9.x atau lebih baru.

b. Browser yang kompatibel untuk melakukan *testing* aplikasi seperti Google Chrome atau Microsoft Edge

c. Tools:

- Vite: Build tool modern yang cepat untuk pengembangan web dengan dukungan module ESM. Versi yang direkomendasikan adalah versi 5.x atau lebih baru.
- React: Library JavaScript untuk membangun antarmuka pengguna berbasis komponen. Versi yang direkomendasikan adalah versi 18.x atau lebih baru.
- TypeScript: Superset JavaScript yang menambahkan tipe statis untuk pengembangan yang lebih aman. Versi yang direkomendasikan adalah versi 5.x atau lebih baru.
- Tailwind CS**S**: Framework CSS utility-first untuk styling cepat dan konsisten. Versi yang direkomendasikan adalah versi 3.x atau lebih baru.
- ShadCN: Kumpulan komponen UI siap pakai yang dapat disesuaikan untuk pengembangan React.

2. Backend

a. Node.js: Platform runtime JavaScript yang diperlukan untuk menjalankan server backend. Versi yang direkomendasikan adalah 20.x atau lebih baru.

b. PostgreSQL: Versi 15.x atau lebih baru disarankan untuk stabilitas dan kinerja optimal. Versi terbaru (misalnya 16.x atau 17.x) membawa fitur dan perbaikan keamanan terbaru, tetapi pastikan aplikasi Anda kompatibel.

c. Hono: Versi yang direkomendasikan: Versi v3.x atau lebih baru. Versi terbaru biasanya mencakup perbaikan bug, peningkatan performa, dan kompatibilitas dengan pembaruan teknologi terbaru.

d. Drizzle ORM: Versi v0.3.x atau lebih baru. Pastikan untuk menggunakan versi stabil yang mendukung fitur seperti migrasi schema dan query builder yang efisien.

e. Middleware: Fungsi atau dependensi yang digunakan untuk menangani berbagai tugas seperti autentikasi, logging, CORS, atau pengelolaan sesi dalam aplikasi backend. Middleware ini dipasang sebagai dependensi dalam aplikasi, seperti:

- Autentikasi: jsonwebtoken untuk menangani autentikasi berbasis token.
- Enkripsi Password: bcrypt untuk melakukan hashing dan verifikasi password.
- Pengaturan Lingkungan: dotenv untuk memuat variabel lingkungan dan konfigurasi aplikasi.
- Pengiriman Email: nodemailer untuk mengirim email otomatis (seperti notifikasi dan verifikasi).

f. Tools tambahan (HonoJS, Drizzle ORM, Middleware.)

C. Struktur Folder

1. Struktur folder proyek secara keseluruhan

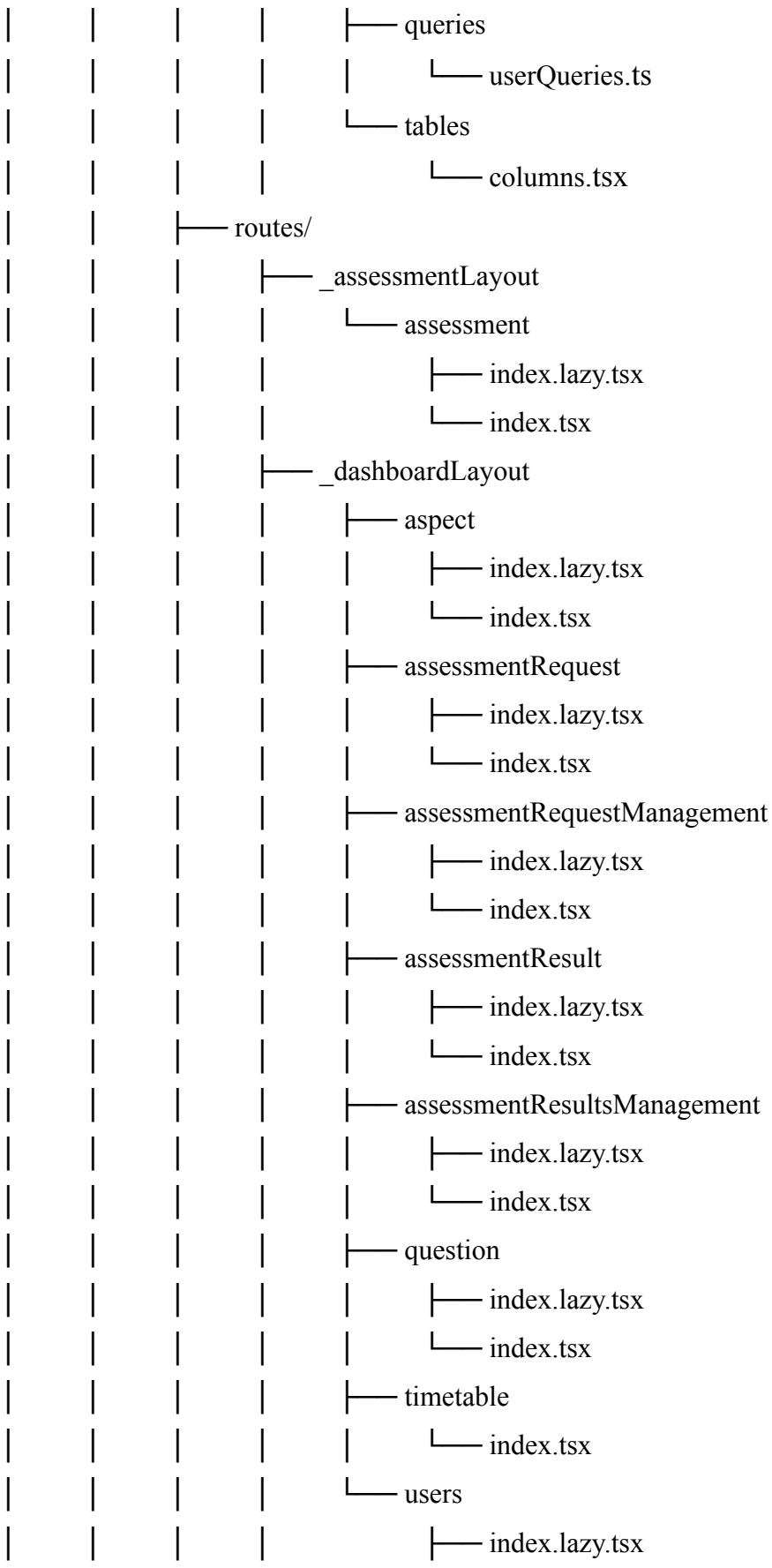
amati

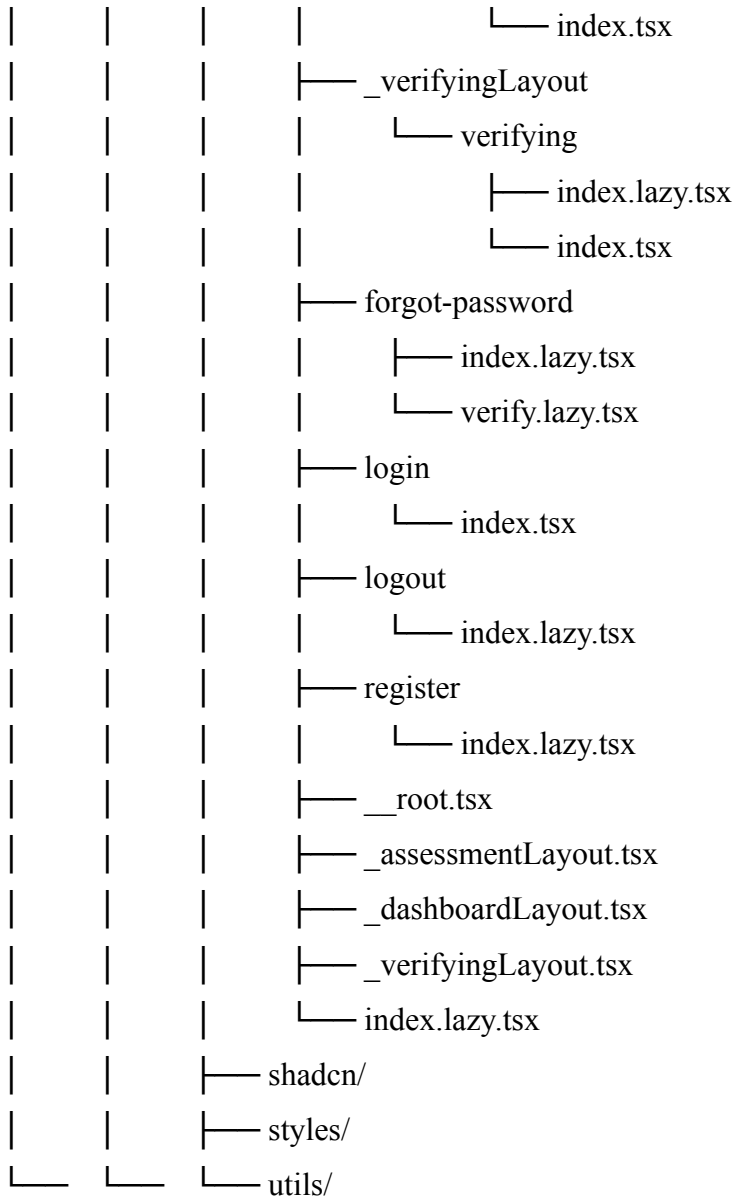
```
├── backend/
│   ├── src/
│   │   ├── drizzle/
│   │   │   ├── migrations/
│   │   │   ├── schema/
│   │   │   │   ├── answerRevisions.ts
│   │   │   │   ├── answers.ts
│   │   │   │   ├── aspects.ts
│   │   │   │   ├── assessments.ts
│   │   │   │   ├── options.ts
│   │   │   │   ├── permissions.ts
│   │   │   │   ├── permissionsToRoles.ts
│   │   │   │   ├── permissionsToUsers.ts
│   │   │   │   ├── questions.ts
│   │   │   │   ├── respondents.ts
│   │   │   │   ├── roles.ts
│   │   │   │   ├── rolesToUsers.ts
│   │   │   │   ├── subAspects.ts
│   │   │   │   └── users.ts
│   │   │   ├── seeds/
│   │   │   │   ├── answersSeeder.ts
```

```
│   │   │   │   ├── aspectsSeeder.ts
│   │   │   │   ├── assessmentsSeeder.ts
│   │   │   │   ├── optionsSeeder.ts
│   │   │   │   ├── permissionSeeder.ts
│   │   │   │   ├── questionSeeder.ts
│   │   │   │   ├── respondentsSeeder.ts
│   │   │   │   ├── rolesSeeder.ts
│   │   │   │   ├── subAspectsSeeder.ts
│   │   │   │   └── userSeeder.ts
│   │   │   ├── index.ts
│   │   │   ├── migration.ts
│   │   │   └── seed.ts
│   │   └──routes/
│   │       ├── assessmentRequest
│   │       │   └── route.ts
│   │       ├── assessmentRequestManagement
│   │       │   └── route.ts
│   │       ├── assessmentResult
│   │       │   └── route.ts
│   │       ├── assessments
│   │       │   └── route.ts
│   │       ├── auth
│   │       │   └── routes.ts
│   │       ├── dashboard
│   │       │   └── route.ts
│   │       ├── dev
│   │       │   └── route.ts
│   │       ├── forgotPassword
│   │       │   └── route.ts
│   │       ├── managementAspect
│   │       │   └── route.ts
│   │       ├── permissions
│   │       │   └── route.ts
```

```
│   │   ├── questions
│   │   │   └── route.ts
│   │   ├── register
│   │   │   └── route.ts
│   │   ├── roles
│   │   │   └── route.ts
│   │   └── users
│   │       └── route.ts
│   └── env/
├── frontend/
│   ├── src/
│   │   ├── components/
│   │   │   ├── styles/
│   │   │   ├── Timetable/
│   │   │   ├── AppHeader.tsx
│   │   │   ├── AppNavbar.tsx
│   │   │   ├── DashboardTable.tsx
│   │   │   ├── NavbarChildMenu.tsx
│   │   │   ├── NavbarMenuItem.tsx
│   │   │   └── PageTemplate.tsx
│   │   ├── modules/
│   │   │   ├── aspectManagement
│   │   │   │   ├── modals
│   │   │   │   │   ├── AspectDeleteModal.tsx
│   │   │   │   │   └── AspectFormModal.tsx
│   │   │   │   └── queries
│   │   │   │       └── aspectQueries.ts
│   │   │   ├── assessmentManagement
│   │   │   │   ├── fileUpload
│   │   │   │   │   └── fileUpload.tsx
│   │   │   │   ├── modals
│   │   │   │   │   ├── ConfirmModal.tsx
│   │   │   │   │   └── FileSizeValidationModal.tsx
```

```
│   │   │   │   │   └── ValidationModal.tsx
│   │   │   │   └── queries
│   │   │   │       └── assessmentQueries.ts
│   │   │   ├── assessmentRequest
│   │   │   │   ├── modals
│   │   │   │   │   ├── ConfirmModal.tsx
│   │   │   │   │   └── CreateAssessmentRequestModal.tsx
│   │   │   │   └── queries
│   │   │   │       └── assessmentRequestQueries.ts
│   │   │   ├── assessmentRequestManagement
│   │   │   │   ├── modals
│   │   │   │   │   └──AssessmentRequest…Modal.tsx
│   │   │   │   └── queries
│   │   │   │       └── assessmentRequest…Queries.ts
│   │   │   ├── assessmentResult
│   │   │   │   └── queries
│   │   │   │       └── assessmentResultQueries.ts
│   │   │   ├── assessmentResultsManagement
│   │   │   │   ├── modals
│   │   │   │   │   └── assessmentResultsFormModal.tsx
│   │   │   │   └── queries
│   │   │   │       └── assessmentResultsManagementQueries.ts
│   │   │   ├── questionsManagement
│   │   │   │   ├── modals
│   │   │   │   │   ├── QuestionDeleteModal.tsx
│   │   │   │   │   └── QuestionFormModal.tsx
│   │   │   │   └── queries
│   │   │   │       └── questionQueries.ts
│   │   │   ├── usersManagement
│   │   │   │   ├── modals
│   │   │   │   │   ├── UserDeleteModal.tsx
│   │   │   │   │   └── UserFormModal.tsx
```


queries
userQueries.ts
tables
columns.tsx
routes/
_assessmentLayout
assessment
index.lazy.tsx
index.tsx
_dashboardLayout
aspect
index.lazy.tsx
index.tsx
assessmentRequest
index.lazy.tsx
index.tsx
assessmentRequestManagement
index.lazy.tsx
index.tsx
assessmentResult
index.lazy.tsx
index.tsx
assessmentResultsManagement
index.lazy.tsx
index.tsx
question
index.lazy.tsx
index.tsx
timetable
index.tsx
users
index.lazy.tsx



2. Penjelasan tiap folder/file utama

a. Frontend:

- src/ → Berisi kode utama frontend menggunakan React
  - components/ → Berisi komponen UI utama aplikasi *frontend*
  - routes/ → Folder untuk menyimpan definisi rute aplikasi menggunakan *@tanstack/react-router.*
  - shadcn/ → Folder untuk komponen UI berbasis ShadCN.

- styles/ → Folder untuk file CSS atau konfigurasi Tailwind CSS.
- utils/ → Berisi utility functions yang membantu logika frontend, seperti pemrosesan data atau pengambilan data dari API.
- modules/ → Berisi berbagai komponen terpisah maupun komponen yang berfungsi untuk integrasi dengan *backend*.

b. Backend:

- src/ → Berisi kode utama untuk server backend.
  - drizzle/ → Folder khusus untuk mengatur konfigurasi Drizzle ORM.
  - migrations/ → Folder untuk menyimpan skrip migrasi database.
  - schema/ → Berisi definisi skema database.
  - seeds/ → Folder untuk menyimpan data awal (seeding) yang akan dimasukkan ke database.
  - index.ts → File utama untuk inisialisasi Drizzle ORM.
  - migration.ts → File untuk mengatur proses migrasi database.
  - seed.ts → File untuk menjalankan seeding ke database.
  - routes/ → Folder untuk mengatur endpoint API dan routing di backend.
  - env → Folder untuk menyimpan konfigurasi atau variabel lingkungan yang digunakan pada backend, seperti koneksi database atau API key.

D. Panduan Instalasi

1. Langkah-langkah untuk setup lingkungan pengembangan

a. Install Node.js

Berikut adalah langkah - langkah untuk pemasangan node.js:

1. Download Node.js

- Kunjungi official website node.js dan unduh versi LTS terbaru (Long Term Support) dari website tersebut.
- Versi LTS ini lebih direkomendasikan karena stabilitasnya.

2. Instalasi Node.js
   - Jalankan installer yang telah diunduh lalu ikuti panduan instalasi dari installer tersebut.
   - Pastikan opsi untuk menambahkan Node.js ke PATH dicentang, agar perintah Node.js dapat digunakan di terminal.
3. Verifikasi Instalasi
   - Buka terminal atau command prompt, lalu jalankan perintah npm -v dan node -v untuk memastikan bahwa Node.js dan NPM sudah terpasang dengan benar pada perangkat.
   - Jika instalasi berhasil, kedua perintah di atas akan menampilkan versi Node.js dan NPM yang telah terpasang pada perangkat.

b. Install PostgreSQL

Berikut adalah langkah - langkah untuk pemasangan PostgreSQL:

1. Download PostgreSQL
   - Kunjungi official website PostgreSQL dan pilih versi PostgreSQL yang sesuai dengan sistem operasi perangkat Anda (Windows, macOS, atau Linux).
   - Untuk pengguna Windows, pilih Windows dan unduh installer dari EnterpriseDB. Untuk pengguna macOS dan Linux, pilih sesuai dengan instruksi di situs tersebut.
2. Instalasi PostgreSQL
   - ➢ Jalankan file installer yang telah diunduh dan ikuti panduan instalasi dari installer tersebut.
   - ➢ Saat proses instalasi, Anda akan diminta untuk memilih opsi seperti:

- Direktori penyimpanan data (default biasanya sudah cukup).
- Port yang digunakan (biasanya menggunakan port 5432, biarkan *default*).
- *Password* untuk *user* 'postgres' (pilih *password* yang aman dan simpan *password* ini karena akan digunakan untuk mengakses PostgreSQL).
- Pilih komponen tambahan (biasanya biarkan opsi default).

➢ Setelah itu, lanjutkan proses instalasi dan tunggu hingga selesai.

3. Verifikasi Instalasi

   Setelah instalasi selesai, buka terminal atau command prompt, dan jalankan perintah berikut untuk memeriksa versi PostgreSQL yang terpasang:

   ```
   psql --version
   ```

   Jika instalasi berhasil, perintah tersebut akan menampilkan versi PostgreSQL yang terpasang di perangkat.

4. Akses PostgreSQL
   - Untuk mengakses PostgreSQL, buka terminal atau *command prompt*, kemudian jalankan perintah berikut:

     ```
     psql -U postgres
     ```

     Setelah itu, Anda akan diminta untuk memasukkan password yang telah Anda terapkan pada saat instalasi.
   - Jika berhasil, Anda akan masuk ke *prompt* PostgreSQL yang dapat digunakan untuk menjalankan *query* SQL.
5. Menggunakan PostgreSQL
   - Untuk membuat database baru, Anda dapat menjalankan perintah SQL berikut di dalam PostgreSQL *prompt*:

     ```
     CREATE DATABASE nama_database;
     ```
   - Untuk keluar dari PostgreSQL prompt, ketikkan perintah: `\q`

c. Install Git

Berikut langkah-langkah pemasangan git ke perangkat:

1. Download Git
    - Kunjungi Git official website dan unduh installer untuk sistem operasi yang digunakan pada perangkat.
2. Instalasi Git
    - Jalankan installer Git dan ikuti panduan instalasinya.
    - Pada bagian "Adjusting your PATH environment", pastikan opsi "Git from the command line" dipilih.
3. Verifikasi Instalasi
    - Buka terminal atau command prompt dan ketik perintah `git --version` untuk memverifikasi instalasi apakah sudah terinstall pada perangkat atau belum.
    - Perintah tersebut akan menampilkan versi Git yang terpasang jika Git telah terpasang pada perangkat.
4. Konfigurasi Awal Git
    - Setelah Git terpasang, lakukan konfigurasi awal dengan menambahkan nama pengguna dan email pada terminal.

d. Install Postman

Berikut adalah cara menginstall Postman pada perangkat:

1. Unduh Postman
    - Kunjungi website resmi postman untuk mengunduh installer postman, sesuaikan dengan sistem operasi perangkat.
    - Setelah mengunduh installer, ikuti panduan instalasi postman tersebut hingga selesai.
2. Membuat Workspace pada Postman
    - Setelah Postman terpasang, buka aplikasi dan buatlah workspace baru dan berikan nama sesuai dengan sistem backend yang akan dilakukan testing (berikan nama AMATI).
    - Tambahkan koleksi API yang ada pada route backend ke Postman untuk dilakukan testing.

3. Penggunaan Environment Postman

   Gunakan fitur environment di Postman untuk menyimpan variabel seperti URL backend (http://localhost:3000) dan token autentikasi yang dapat digunakan dalam pengujian API.

e. Clone repository

   Clone kode proyek dari repositori Git dibawah ini lalu inputkan perintah tersebut pada terminal vscode (jangan lupa untuk menentukan direktorinya terlebih dahulu). (git clone (https://github.com/digitalsolutiongroup/amati.git)

f. Setup frontend

   Berikut adalah langkah - langkah untuk setup frontend:

   1. Pindah ke folder proyek frontend:
      `cd nama-folder-proyek`
   2. Install Dependencies Menggunakan npm
      - Setelah berada di dalam folder proyek, jalankan perintah berikut untuk menginstal semua dependensi yang dibutuhkan untuk proyek frontend:
        `npm install`
      - Perintah ini akan membaca file `package.json` dan mengunduh semua paket yang terdaftar dalam `dependencies` dan `devDependencies` ke dalam folder `node_modules`.
   3. Install Dependencies Menggunakan pnpm
      - Jika Anda menggunakan pnpm sebagai package manager, pastikan pnpm sudah terpasang di sistem Anda. Jika belum, Anda bisa menginstalnya dengan menjalankan perintah berikut:
        `npm install -g pnpm`
      - Setelah itu, untuk menginstal dependensi proyek menggunakan pnpm, jalankan perintah berikut:
        `pnpm install`
   4. Konfigurasi .env file

- Salin file .env.example dengan nama .env di dalam folder frontend.

5. Verifikasi Instalasi
    - Setelah proses instalasi selesai, pastikan tidak ada `error` dan semua dependensi berhasil terpasang. Anda dapat memeriksa apakah folder `node_modules` sudah ada dan berisi paket-paket yang diperlukan.
    - Untuk memastikan bahwa proyek dapat dijalankan, Anda dapat memulai aplikasi dengan perintah:
      `npm start`
      jika menggunakan pnpm:
      `pnpm start`

g. Setup backend
    1. Buka terminal dan pindah ke folder proyek backend:
       `cd nama-folder-backend`
    2. Install Dependencies Menggunakan npm
        - Setelah berada di dalam folder proyek, jalankan perintah berikut untuk menginstal semua dependensi yang dibutuhkan untuk proyek backend:
          `npm install`
        - Perintah ini akan membaca file `package.json` dan mengunduh semua paket yang terdaftar dalam `dependencies` dan `devDependencies` ke dalam folder `node_modules`.
    3. Install Dependencies Menggunakan pnpm
        - Jika Anda menggunakan pnpm sebagai package manager, pastikan pnpm sudah terpasang di sistem Anda. Jika belum, Anda bisa menginstalnya dengan menjalankan perintah berikut:
          `npm install -g pnpm`
        - Setelah itu, untuk menginstal dependensi proyek menggunakan pnpm, jalankan perintah berikut:

```
pnpm install
```

4. Konfigurasi .env file
    - Salin file .env.example dengan nama .env di dalam folder backend.
    - Kemudian ubahlah isi file .env menjadi seperti berikut:

```
BASE_URL = http://localhost:5173/
APP_PORT = 3000

DATABASE_URL =
postgres://postgres:12345@localhost:54
32/amati-test

ACCESS_TOKEN_SECRET = "asdfghjkl"
REFRESH_TOKEN_SECRET = "asdfghjkl"
RESET_PASSWORD_TOKEN_SECRET="asdfghjkl
"
COOKIE_SECRET = "asdfghjkl"

SMTP_USERNAME="accsuk8@gmail.com"
SMTP_PASSWORD="vkjc fpfn ylue kwhm"
SMTP_HOST="smtp.gmail.com"
SMTP_PORT=587
```

h. Migrasi database
    1. Buka terminal dan pindah ke folder backend proyek:
       ```
       cd apps/backend
       ```
    2. Migrasi Tabel ke Database:
        - Untuk melakukan migrasi tabel (membuat struktur tabel pada database), jalankan perintah berikut:
          ```
          pnpm run db:push
          ```
        - Perintah ini akan menjalankan migrasi schema dan membuat tabel-tabel yang diperlukan di dalam database sesuai dengan definisi yang ada pada kode.
    3. Migrasi Data (Seeder) ke Database:
        - Untuk mengisi database dengan data awal (seeding), jalankan perintah berikut:
          ```
          pnpm run db:seed
          ```
        - Perintah ini akan menjalankan proses seeding untuk memasukkan data awal atau contoh ke dalam tabel

database, seperti pengguna, role, atau data penting lainnya yang dibutuhkan oleh aplikasi.

2. **Perintah untuk menjalankan aplikasi**

    a. npm run dev (frontend)

    ➢ Menjalankan Aplikasi Frontend:

    - Buka terminal dan pindah ke folder frontend proyek: `cd apps/frontend`
    - Untuk menjalankan aplikasi frontend dalam mode pengembangan, jalankan perintah berikut di terminal: `npm run dev`
    - Perintah ini akan menjalankan server pengembangan dan aplikasi frontend dapat diakses melalui `http://localhost:5173` (atau port yang telah dikonfigurasi).

    b. npm run start (backend)

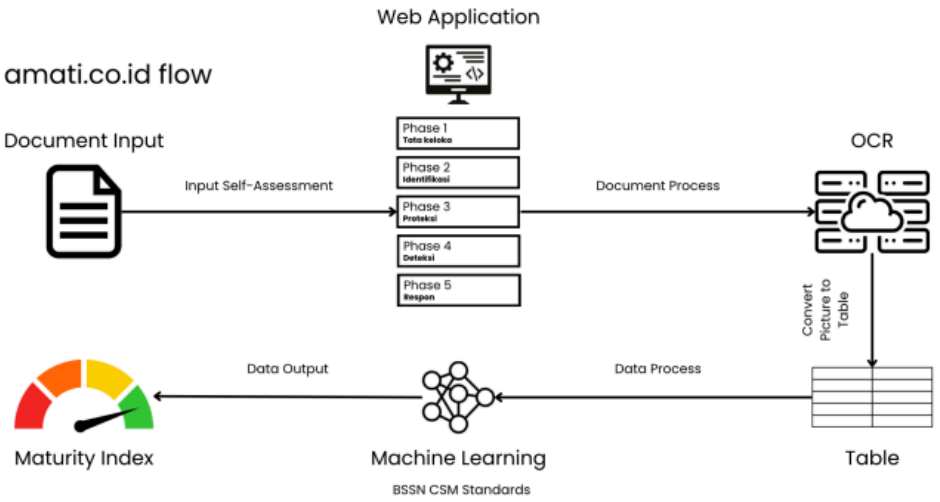
    ➢ Menjalankan Aplikasi Backend:

    - Buka terminal dan pindah ke folder backend proyek: `cd apps/backend`
    - Untuk menjalankan aplikasi frontend dalam mode pengembangan, jalankan perintah berikut di terminal: `npm run start`
    - Perintah ini akan menjalankan server pengembangan dan aplikasi frontend dapat diakses melalui `http://localhost:3000` (atau port yang telah dikonfigurasi).

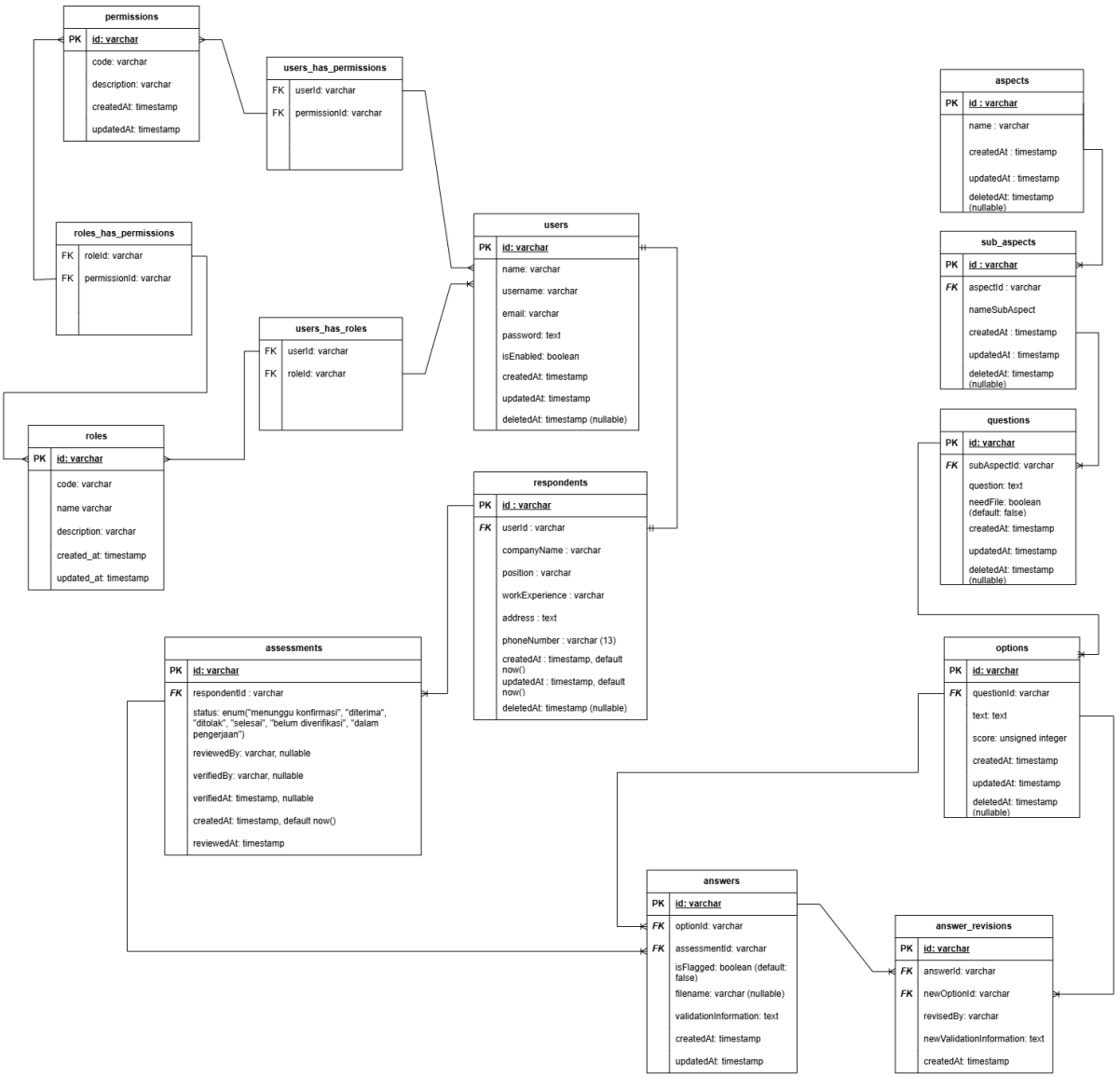
E. Arsitektur Aplikasi

Website AMATI mendukung dua *role user* yaitu *superadmin*, dan responden. *Superadmin* dapat mengelola berbagai data, mulai dari responden, menambahkan aspek, sub aspek dan daftar pertanyaan beserta pilihan jawabannya. Sedangkan responden dapat mengajukan permohonan asesmen, mengerjakan soal asesmen, melihat *history* dan hasil pengerjaan latihan.

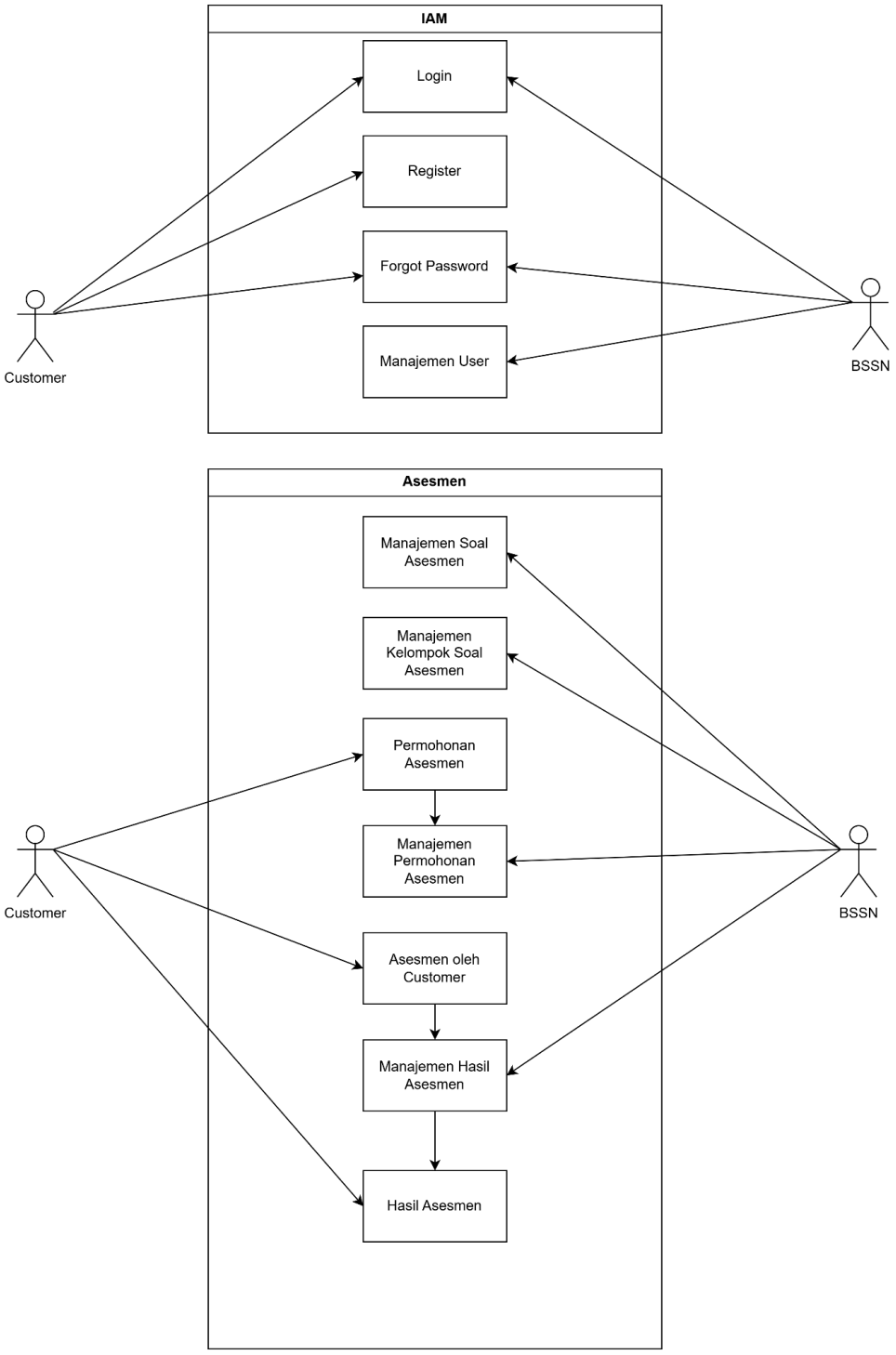
1. **Diagram arsitektur**



Gambar 1. Diagram Arsitektur



Gambar 2. Struktur *database*



Gambar 3. *Use case diagram*

2. **Penjelasan alur kerja utama (request-response)**

   a. POST /auth/register

   Fungsi: Mendaftarkan pengguna baru.

   Request

Headers	Jenis konten yang dikirimkan dalam request.	`Content-Type: application/json`
Body	Data yang dikirimkan oleh klien untuk mendaftarkan pengguna baru.	`{ "username": "john_doe", "email": "john@example.com", "password": "password123" }`

Tabel 2.1 Register API

Response

Headers	Jenis konten yang diterima dalam response.	`Content-Type: application/json`
Body	Data yang dikembalikan server setelah proses registrasi berhasil. Biasanya berisi token otentikasi atau pesan sukses.	`{ "message": "User registered successfully", "status": "success" }`

Tabel 2.2 Register API

F. Pemasangan dan Konfigurasi Frontend

1. **Framework dan library utama (React, TanStack Router, Tailwind CSS, dll.)**

   a. Inisialisasi Proyek *Frontend*

   - Jika proyek belum diinisialisasi, buat proyek baru dengan perintah berikut:

     ```
     npm create vite@latest frontend-app
     cd frontend-app
     ```

   - Pilih React sebagai framework dan JavaScript/TypeScript sesuai kebutuhan.

b. Instalasi Framework dan Library Utama

Pasang *library* utama yang dibutuhkan dalam proyek *frontend*:

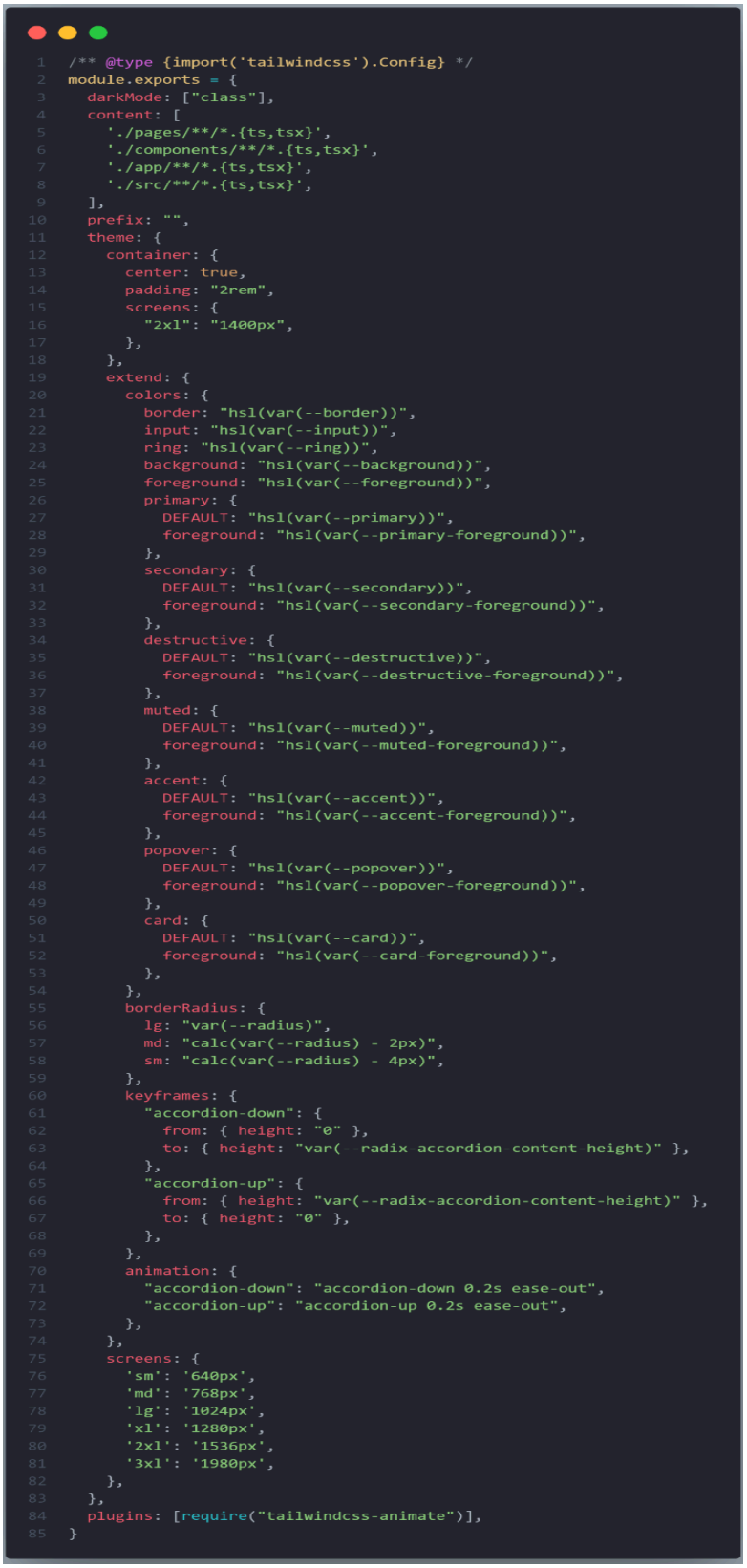
```
npm install react react-dom @tanstack/router tailwindcss @shadcn/ui
```

Penjelasan Library:

- React: *Framework* utama untuk membangun antarmuka pengguna.
- TanStack Router: *Library* untuk menangani navigasi halaman dengan pendekatan yang lebih fleksibel.
- Tailwind CSS: *Framework* CSS yang memungkinkan pengembangan UI dengan cepat melalui utilitas kelas.
- ShadCN UI: Komponen UI siap pakai untuk integrasi dengan Tailwind CSS.

c. Konfigurasi Tailwind CSS

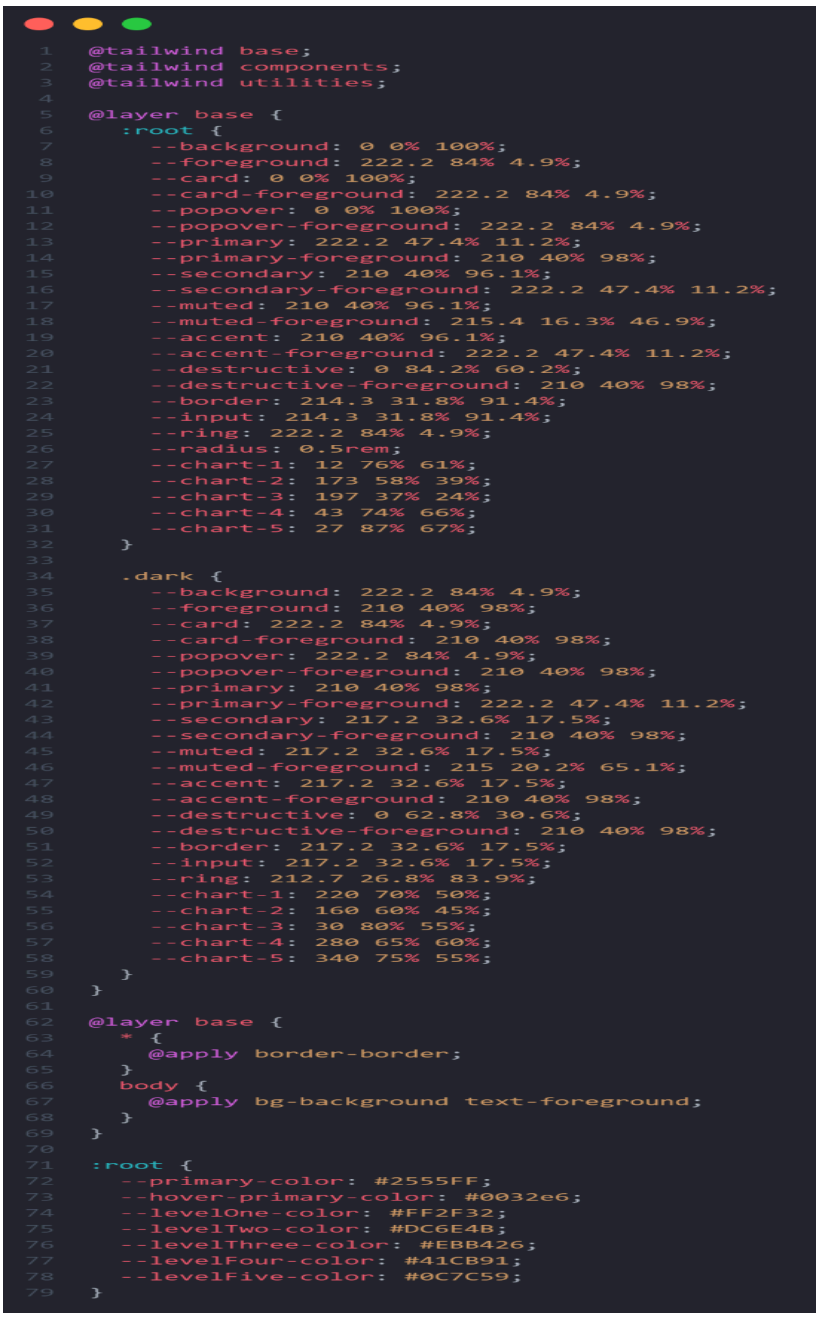
- Inisialisasi Tailwind CSS dengan perintah:
  `npx tailwindcss init -p`
- Edit file `tailwind.config.js` dan tambahkan konfigurasi berikut:



```
/** @type {import('tailwindcss').Config} */
module.exports = {
  darkMode: ["class"],
  content: [
    './pages/**/*.{ts,tsx}',
    './components/**/*.{ts,tsx}',
    './app/**/*.{ts,tsx}',
    './src/**/*.{ts,tsx}',
  ],
  prefix: "",
  theme: {
    container: {
      center: true,
      padding: "2rem",
      screens: {
        "2xl": "1400px",
      },
    },
    extend: {
      colors: {
        border: "hsl(var(--border))",
        input: "hsl(var(--input))",
        ring: "hsl(var(--ring))",
        background: "hsl(var(--background))",
        foreground: "hsl(var(--foreground))",
        primary: {
          DEFAULT: "hsl(var(--primary))",
          foreground: "hsl(var(--primary-foreground))",
        },
        secondary: {
          DEFAULT: "hsl(var(--secondary))",
          foreground: "hsl(var(--secondary-foreground))",
        },
        destructive: {
          DEFAULT: "hsl(var(--destructive))",
          foreground: "hsl(var(--destructive-foreground))",
        },
        muted: {
          DEFAULT: "hsl(var(--muted))",
          foreground: "hsl(var(--muted-foreground))",
        },
        accent: {
          DEFAULT: "hsl(var(--accent))",
          foreground: "hsl(var(--accent-foreground))",
        },
        popover: {
          DEFAULT: "hsl(var(--popover))",
          foreground: "hsl(var(--popover-foreground))",
        },
        card: {
          DEFAULT: "hsl(var(--card))",
          foreground: "hsl(var(--card-foreground))",
        },
      },
      borderRadius: {
        lg: "var(--radius)",
        md: "calc(var(--radius) - 2px)",
        sm: "calc(var(--radius) - 4px)",
      },
      keyframes: {
        "accordion-down": {
          from: { height: "0" },
          to: { height: "var(--radix-accordion-content-height)" },
        },
        "accordion-up": {
          from: { height: "var(--radix-accordion-content-height)" },
          to: { height: "0" },
        },
      },
      animation: {
        "accordion-down": "accordion-down 0.2s ease-out",
        "accordion-up": "accordion-up 0.2s ease-out",
      },
    },
    screens: {
      'sm': '640px',
      'md': '768px',
      'lg': '1024px',
      'xl': '1280px',
      '2xl': '1536px',
      '3xl': '1980px',
    },
  },
  plugins: [require("tailwindcss-animate")],
}
```

Gambar 4. Konfigurasi Tailwind CSS

- Tambahkan direktif Tailwind CSS ke dalam file src/index.css:



```
@tailwind base;
@tailwind components;
@tailwind utilities;

@layer base {
  :root {
    --background: 0 0% 100%;
    --foreground: 222.2 84% 4.9%;
    --card: 0 0% 100%;
    --card-foreground: 222.2 84% 4.9%;
    --popover: 0 0% 100%;
    --popover-foreground: 222.2 84% 4.9%;
    --primary: 222.2 47.4% 11.2%;
    --primary-foreground: 210 40% 98%;
    --secondary: 210 40% 96.1%;
    --secondary-foreground: 222.2 47.4% 11.2%;
    --muted: 210 40% 96.1%;
    --muted-foreground: 215.4 16.3% 46.9%;
    --accent: 210 40% 96.1%;
    --accent-foreground: 222.2 47.4% 11.2%;
    --destructive: 0 84.2% 60.2%;
    --destructive-foreground: 210 40% 98%;
    --border: 214.3 31.8% 91.4%;
    --input: 214.3 31.8% 91.4%;
    --ring: 222.2 84% 4.9%;
    --radius: 0.5rem;
    --chart-1: 12 76% 61%;
    --chart-2: 173 58% 39%;
    --chart-3: 197 37% 24%;
    --chart-4: 43 74% 66%;
    --chart-5: 27 87% 67%;
  }

  .dark {
    --background: 222.2 84% 4.9%;
    --foreground: 210 40% 98%;
    --card: 222.2 84% 4.9%;
    --card-foreground: 210 40% 98%;
    --popover: 222.2 84% 4.9%;
    --popover-foreground: 210 40% 98%;
    --primary: 210 40% 98%;
    --primary-foreground: 222.2 47.4% 11.2%;
    --secondary: 217.2 32.6% 17.5%;
    --secondary-foreground: 210 40% 98%;
    --muted: 217.2 32.6% 17.5%;
    --muted-foreground: 215 20.2% 65.1%;
    --accent: 217.2 32.6% 17.5%;
    --accent-foreground: 210 40% 98%;
    --destructive: 0 62.8% 30.6%;
    --destructive-foreground: 210 40% 98%;
    --border: 217.2 32.6% 17.5%;
    --input: 217.2 32.6% 17.5%;
    --ring: 212.7 26.8% 83.9%;
    --chart-1: 220 70% 50%;
    --chart-2: 160 60% 45%;
    --chart-3: 30 80% 55%;
    --chart-4: 280 65% 60%;
    --chart-5: 340 75% 55%;
  }
}

@layer base {
  * {
    @apply border-border;
  }
  body {
    @apply bg-background text-foreground;
  }
}

:root {
  --primary-color: #2555FF;
  --hover-primary-color: #0032e6;
  --levelOne-color: #FF2F32;
  --levelTwo-color: #DC6E4B;
  --levelThree-color: #EBB426;
  --levelFour-color: #41CB91;
  --levelFive-color: #0C7C59;
}
```

Gambar 5. Direktif Tailwind CSS

d. Konfigurasi Tanstack Router

- Buat file router di `src/App.tsx`:

```tsx
import { RouterProvider, createRouter } from "@tanstack/react-router";
import { QueryClient, QueryClientProvider } from "@tanstack/react-query";

import { MantineProvider } from "@mantine/core";
import { Notifications } from "@mantine/notifications";

import { routeTree } from "./routeTree.gen";

import "@mantine/core/styles.css";
import "@mantine/notifications/styles.css";
import "@mantine/dates/styles.css";
import { AuthProvider } from "./contexts/AuthContext";

const queryClient = new QueryClient();

const router = createRouter({
    routeTree,
    context: { queryClient: queryClient },
    defaultPreloadStaleTime: 0,
});

declare module "@tanstack/react-router" {
    interface Register {
        router: typeof router;
    }
}

function App() {
    return (
        <MantineProvider>
            <Notifications />
            <QueryClientProvider client={queryClient}>
                <AuthProvider>
                    <RouterProvider router={router} />
                </AuthProvider>
            </QueryClientProvider>
        </MantineProvider>
    );
}

export default App;
```

Gambar 6. Konfigurasi Tanstack Router

- Integrasikan router di `src/main.tsx`:

```tsx
import React from "react";
import ReactDOM from "react-dom/client";
import App from "./App.tsx";
import "./index.css";
import "./styles/tailwind.css";
import "./styles/fonts/inter.css";

ReactDOM.createRoot(document.getElementById("root")!).render(
    <React.StrictMode>
        <App />
    </React.StrictMode>
);
```

Gambar 7. Integrasi *Router*

e. Konfigurasi ShadCN UI

- Inisialisasi ShadCN UI:

```
npx shadcn-ui@latest init
```

- Ikuti instruksi untuk memilih konfigurasi yang sesuai dengan proyek.
- Tambahkan komponen ShadCN yang dibutuhkan, contohnya:
  ```
  npx shadcn-ui@latest add button
  ```

2. Cara menambahkan dependensi baru

Menambahkan dependensi ke dalam proyek sangat penting untuk mengintegrasikan fitur atau alat tambahan yang dibutuhkan dalam pengembangan aplikasi. Berikut adalah langkah-langkahnya:

a. Menambahkan Dependensi Baru (Dependencies)

Dependensi adalah paket yang dibutuhkan untuk menjalankan aplikasi di lingkungan produksi.

- Menggunakan npm:
  ```
  npm install nama-paket
  ```
- Menggunakan pnpm:
  ```
  pnpm add nama-paket
  ```

b. Menambahkan Dependensi untuk Pengembangan (DevDependencies)

DevDependencies adalah paket yang hanya dibutuhkan saat pengembangan, bukan saat aplikasi berjalan di lingkungan produksi.

- Menggunakan npm:
  ```
  npm install nama-paket --save-dev
  ```
- Menggunakan pnpm:
  ```
  pnpm add nama-paket -D
  ```

c. Memeriksa Dependensi yang Terpasang

- Setelah instalasi selesai, pastikan dependensi berhasil ditambahkan:
  ```
  npm list nama-paket
  ```
- Atau untuk melihat semua dependensi:
  ```
  npm list
  ```
- Jika menggunakan pnpm:
  ```
  pnpm list
  ```

d. Menghapus Dependensi (Jika Diperlukan)

- Menggunakan npm:

```
npm uninstall nama-paket
```

- Menggunakan pnpm:

```
pnpm remove nama-paket
```

e. Catatan Penting

- Pastikan dependensi yang ditambahkan sesuai dengan versi dan kompatibilitas proyek Anda.
- Periksa file package.json untuk memastikan dependensi baru telah ditambahkan dengan benar.
- Jika ada masalah, coba hapus folder node_modules dan file package-lock.json/pnpm-lock.yaml, lalu instal ulang dengan:

```
npm install
```

Atau

```
pnpm install
```

3. Cara membuat komponen baru

Berikut adalah langkah-langkah untuk membuat komponen baru di aplikasi React:

a. Tentukan Struktur Direktori

Sebelum membuat komponen baru, pastikan Anda memiliki struktur folder yang terorganisir. Biasanya, komponen ditempatkan di dalam folder `src/components/`. Misalnya:

```
├── src/
│   ├── components/
│   │   ├── header/
│   │   │   └── header.tsx
│   │   ├── footer/
│   │   │   └── footer.tsx
```

b. Membuat Komponen Fungsi dengan TypeScript

➢ Untuk membuat komponen baru menggunakan TypeScript, pastikan file komponen memiliki ekstensi `.tsx` (TypeScript dengan JSX).

1. Buat File Komponen:

Misalnya, untuk membuat komponen Header, buat file Header.tsx di dalam folder `components/header/`

2. Tulis Kode Komponen:

   Berikut adalah contoh komponen fungsi dengan TypeScript:

```tsx
import React from 'react';

// Definisikan tipe untuk props (jika ada)
interface HeaderProps {
  title: string;
}

const Header: React.FC<HeaderProps> = ({ title }) => {
  return (
    <header>
      <h1>{title}</h1>
    </header>
  );
};

export default Header;
```

Gambar 8. Contoh Komponen

Pada contoh di atas, kita menggunakan `interface` untuk mendefinisikan tipe dari props yang diterima oleh komponen. `React.FC` (Functional Component) digunakan untuk mendeklarasikan komponen fungsional di React dengan TypeScript.

3. Menggunakan Komponen di File Lain

   Setelah membuat komponen baru, Anda dapat menggunakannya di bagian lain aplikasi dengan cara yang mirip dengan JavaScript biasa.

   a. Impor Komponen

      Di file tempat Anda ingin menggunakan komponen (`App.tsx` atau lainnya), impor komponen yang baru dibuat.

```tsx
import Header from './components/Header/Header';
```

Gambar 9. Impor Komponen

b. Tambahkan Komponen dalam JSX

Gunakan komponen di dalam JSX dengan cara berikut:

```jsx
import React from 'react';
import Header from './components/Header/Header';

const App: React.FC = () => {
  return (
    <div>
      <Header title="Selamat datang di aplikasi saya!" />
      <p>Konten lainnya...</p>
    </div>
  );
};

export default App;
```

Gambar 10. Komponen dalam JSX

c. Menjalankan Aplikasi:

Setelah membuat komponen, jalankan aplikasi untuk memastikan semuanya berfungsi dengan baik:

`npm run dev`

Atau

`pnpm run dev`

4. Best practices pengembangan frontend

Berikut adalah praktik terbaik dalam pengembangan frontend untuk menjaga kode tetap bersih, mudah dipahami, dan mudah dipelihara:

a. Struktur Proyek yang Terorganisir

- Pisahkan komponen berdasarkan fungsionalitas (misalnya: `components`, `pages`, `assets`, `hooks`).
- Gunakan penamaan yang konsisten untuk file dan folder (`PascalCase` untuk komponen, `camelCase` untuk fungsi dan variabel).

Contoh Struktur Direktori:

```
├── src/
│   │   ├── components/
```

```
│   │   ├── Header/
│   │   │   ├── Header.tsx
│   │   │   └── Header.module.css
│   ├── pages/
│   │   ├── Home/
│   │   │   └── Home.tsx
│   ├── hooks/
│   │   ├── Home/
│   │   │   └── useFetchData.ts
│   ├── assets/
│   │   ├── images/
│   │   ├── styles/
```

b. Gunakan Komponen yang *Reusable*

- Pecah UI menjadi komponen kecil yang dapat digunakan kembali.
- Hindari komponen dengan terlalu banyak fungsi (lebih dari satu tanggung jawab).

Contoh Komponen *Reusable*:

```tsx
import React from 'react';

interface ButtonProps {
  label: string;
  onClick: () => void;
}

const Button: React.FC<ButtonProps> = ({ label, onClick }) => {
  return <button onClick={onClick}>{label}</button>;
};

export default Button;
```

Gambar 11. Komponen *Reusable*

c. Kelola *State* dengan Bijak

- Gunakan *local state* untuk data yang hanya digunakan di dalam komponen.
- Gunakan *global state management* (misalnya: *Zustand*, *Redux*) untuk data yang digunakan di berbagai komponen.

d. Konsistensi dalam *Slicing*

- Gunakan satu pendekatan *styling* (misalnya: Tailwind CSS, *CSS Modules*, atau *Styled Components*).
- Hindari campuran berbagai metode *styling* di satu proyek.

Contoh *Styling* dengan Tailwind CSS:

```
<button className="bg-blue-500 text-white px-4 py-2 rounded-md">
  Click Me
</button>
```

Gambar 12. Contoh *Styling* dengan Tailwind CSS

e. Validasi *Input* dan *Error Handling*

- Selalu validasi *input* dari pengguna di *frontend*.
- Tampilkan pesan kesalahan yang jelas dan ramah pengguna.

Contoh Validasi *Form*:

```
const handleSubmit = (e: React.FormEvent) => {
    e.preventDefault();
    if (!inputValue) {
        alert('Input tidak boleh kosong!');
    }
};
```

Gambar 13. Contoh Validasi *Form*

f. Optimalkan Kinerja (*performance*)

- Gunakan *lazy loading* untuk komponen yang jarang digunakan.
- Hindari *re-rendering* yang tidak perlu dengan `React.memo` atau `useMemo`.
- Gunakan *caching* untuk data API dengan *library* seperti *TanStack Query*.

g. Lakukan *Testing*

- Gunakan *unit testing* untuk komponen penting (misalnya: Jest, React Testing Library).
- Uji komponen utama, fungsi penting, dan skenario kritis.

Contoh Pengujian dengan Jest:

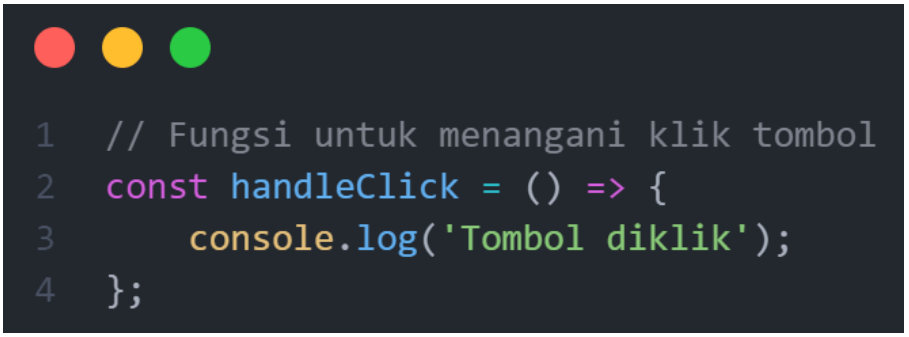
```
test('renders button with label', () => {
    render(<Button label="Click Me" onClick={() => { }} />);
    expect(screen.getByText('Click Me')).toBeInTheDocument();
});
```

Gambar 14. Contoh Pengujian dengan Jest

h. Dokumentasikan Kode

- Tambahkan komentar pada bagian kode yang kompleks.
- Gunakan *tools* seperti *Storybook* untuk dokumentasi komponen UI.

Contoh Komentar:



```
// Fungsi untuk menangani klik tombol
const handleClick = () => {
    console.log('Tombol diklik');
};
```

Gambar 15. Contoh Komentar

i. Gunakan Git dengan Baik

- Buat *branch* baru untuk setiap fitur (`feature/fitur-xyz`).
- Gunakan *commit* yang deskriptif (`fix: perbaikan bug pada komponen X`).
- Lakukan *pull request* dan *code review* secara rutin.

j. Ikuti Standar Kode (*Linting & Formatting*)

- Gunakan ESLint untuk menjaga kualitas kode.
- Gunakan Prettier untuk memastikan format kode konsisten di seluruh proyek.

Instalasi ESLint & Prettier:

```
npm install eslint prettier --save-dev
```

5. *Testing* dan *Debugging*

Testing dan *debugging* merupakan tahap penting dalam pengembangan *frontend* untuk memastikan aplikasi berjalan dengan baik dan bebas dari *bug*.

a. *Unit Testing*

- *Library* yang direkomendasikan: Jest, React Testing Library.
- Contoh perintah untuk menjalankan *unit test*:

  `npm run test`

  Contoh kasus *unit test* sederhana:

```tsx
// Button.test.tsx
import { render, screen } from '@testing-library/react';
import Button from './Button';

test('menampilkan teks tombol dengan benar', () => {
  render(<Button label="Klik Saya" />);
  expect(screen.getByText('Klik Saya')).toBeInTheDocument();
});
```

Gambar 16. Contoh *Unit Test* Sederhana

b. *Integration Testing*

Integration testing memeriksa apakah berbagai bagian aplikasi bekerja bersama dengan baik.

- Contoh perintah untuk menjalankan *integration test*:

  `npm run test:integration`

- Gunakan Jest dan *Testing Library* untuk menguji integrasi antar komponen.

c. *End-to-End (E2E) Testing*

E2E testing memeriksa aplikasi secara keseluruhan, termasuk interaksi pengguna di antarmuka.

- Tool yang direkomendasikan: Cypress, Playwright.
- Contoh perintah untuk menjalankan *E2E test*:

  `npm run test:e2e`

d. *Debugging*

Debugging digunakan untuk menemukan dan memperbaiki kesalahan pada aplikasi.

➢ Gunakan alat *debugging* bawaan *browser*:

- *Chrome DevTools: Inspect, Console, Network.*

➢ Tambahkan *breakpoint* di kode:

```
console.log('Debugging variable:',
variable);
```

➢ Gunakan ekstensi untuk *debugging* di IDE:

- *Visual Studio Code*: *Debugger for Chrome*.

e. *Error Handling*

Pastikan aplikasi menangani kesalahan dengan baik.

Gunakan *boundary error* pada komponen React:

```
class ErrorBoundary extends React.Component {
    state = { hasError: false };

    static getDerivedStateFromError(error: Error) {
        return { hasError: true };
    }

    render() {
        if (this.state.hasError) {
            return <h1>Terjadi kesalahan. Silakan coba lagi.</h1>;
        }
        return this.props.children;
    }
}
```

Gambar 17. *Boundary Error* pada Komponen React

f. *Best Practices* untuk *Testing* dan *Debugging*

- Tulis *test* yang mudah dipahami dan dipelihara.
- Fokus pada *edge cases* dan *critical path*s.
- Jalankan *test* secara otomatis sebelum *deployment*.
- Gunakan *coverage tools* untuk memastikan kode yang diuji mencakup bagian penting aplikasi.

G. Pemasangan dan Konfigurasi *Backend*

1. Framework dan library utama (HonoJS, Drizzle ORM, dll.)

a. Instalasi HonoJS:

```
npm install hono
```

b. Instalasi Drizzle ORM:

```
npm install drizzle-orm
```

2. Struktur routing dan middleware

a. Contoh Routing di roles/route.ts

```
import { Hono } from "hono";
import db from "../../drizzle";
import { rolesSchema } from "../../drizzle/schema/roles";

const rolesRoute = new Hono()
    //get all permissions
    .get("/", async (c) => {
        const roles = await db
            .select({
                id: rolesSchema.id,
                code: rolesSchema.code,
                name: rolesSchema.name,
            })
            .from(rolesSchema);

        return c.json(roles);
    });

export default rolesRoute;
```

Gambar 18. Routing roles pada Backend

b. Integrasi Routing di index.ts

```
const app = new Hono<HonoEnv>();

const routes = app
    .use(logger())
    .use(
        cors({
            origin: "*",
        })
    )
    .use(async (c, next) => {
        const cookieSecret = appEnv.COOKIE_SECRET;

        if (!cookieSecret)
            throw new HTTPException(500, {
                message: "The 'COOKIE_SECRET' env is not set",
            });

        const accessToken = await getSignedCookie(
            c,
            cookieSecret,
            "access_token",
            "secure"
        );

        if (accessToken) {
            const payload = await verifyAccessToken(accessToken);

            if (payload) c.set("uid", payload.uid);
        } else {
            const authHeader = c.req.header("Authorization");

            if (authHeader && authHeader.startsWith("Bearer ")) {
                const token = authHeader.substring(7);
                const payload = await verifyAccessToken(token);

                if (payload) c.set("uid", payload.uid);
            }
        }

        await next();
    })
    .use(async (c, next) => {
        console.log("Incoming request:", c.req.path);
        await next();
        console.log("Outgoing response:", c.res.status);
        if (c.res.status !== 200) {
            console.log(await c.res.text());
        }
    })
    .get("/test", (c) => {
        return c.json({
            message: "Server is up",
        } as const);
    })
    .route("/auth", authRoutes)
    .route("/users", usersRoute)
    .route("/permissions", permissionRoutes)
    .route("/dashboard", dashboardRoutes)
    .route("/roles", rolesRoute)
    .route("/dev", devRoutes)
    .route("/questions", questionsRoute)
    .route("/management-aspect", managementAspectsRoute)
    .route("/register", respondentsRoute)
    .route("/assessmentResult", assessmentResultRoute)
    .route("/assessmentRequest", assessmentRequestRoute)
    .route("/forgot-password", forgotPasswordRoutes)
    .route("/assessments", assessmentsRoute)
    .route("/assessmentRequestManagement",assessmentsRequestManagementRoutes)
    .onError((err, c) => {
        if (err instanceof DashboardError) {
            return c.json(
                {
                    message: err.message,

                    errorCode: err.errorCode,
                    formErrors: err.formErrors,
                },
                err.statusCode
            );
        }
        if (err instanceof HTTPException) {
            console.log(err);
            return c.json(
                {
                    message: err.message,
                },
                err.status
            );
        } else {
            console.error(err);
            return c.json(
                {
                    message:
                        "Something is wrong in our side. We're working to fix it",
                },
                500
            );
        }
    });

const port = appEnv.APP_PORT;
console.log(`Server is running on port ${port}`);
```

Gambar 19. Integrasi Routing pada Backend

c. Middleware

- Middleware authInfo

```
const authInfo = createMiddleware<HonoEnv>(async (c, next) => {
    const { uid, currentUser } = c.var;

    // Proceed only if uid is present and currentUser is not already set
    if (uid && !currentUser) {
        const user = await db
            .select()
            .from(users)
            .where(
                and(
                    eq(users.id, uid),
                    eq(users.isEnabled, true),
                    isNull(users.deletedAt)
                )
            )
            .leftJoin(
                permissionsToUsers,
                eq(permissionsToUsers.userId, users.id)
            )
            .leftJoin(rolesToUsers, eq(rolesToUsers.userId, users.id))
            .leftJoin(rolesSchema, eq(rolesToUsers.roleId, rolesSchema.id))
            .leftJoin(
                permissionsToRoles,
                eq(permissionsToRoles.roleId, rolesSchema.id)
            )
            .leftJoin(
                permissionsSchema,
                or(
                    eq(permissionsSchema.id, permissionsToUsers.permissionId),
                    eq(permissionsSchema.id, permissionsToRoles.permissionId)
                )
            );

        const roles = new Set<RoleCode>();
        const permissions = new Set<SpecificPermissionCode>();

        user.forEach((user) => {
            if (user.roles?.code) {
                roles.add(user.roles.code as RoleCode);
            }

            if (user.permissions?.code) {
                permissions.add(
                    user.permissions.code as SpecificPermissionCode
                );
            }
        });

        // Setting the currentUser with fetched data
        c.set("currentUser", {
            id: user[0].users.id, // Adding user ID here
            name: user[0].users.name, // Assuming the first result is the user
            permissions: Array.from(permissions),
            roles: Array.from(roles),
        });
    }

    await next();
});

export default authInfo;
```

Gambar 20. Middleware Auth Info pada Struktur Middleware

Fungsi: Digunakan untuk memuat dan mengatur informasi autentikasi untuk pengguna. Middleware ini meminta database untuk mengambil detail pengguna, peran, dan izin berdasarkan ID pengguna yang disediakan dalam konteks. Middleware menggabungkan beberapa tabel untuk mengumpulkan semua data yang relevan secara efisien, dan kemudian memproses data ini untuk melampirkan peran dan izin

pengguna ke dalam konteks. Jika pengguna ditemukan dan aktif, detailnya ditambahkan ke konteks untuk digunakan dalam middleware atau route berikutnya.

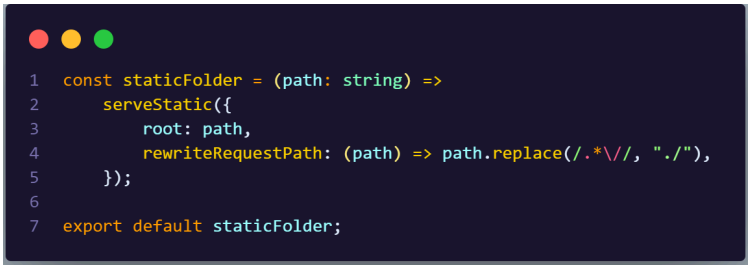
- Middleware checkPermission

```
const checkPermission = (...permissions: PermissionCode[]) =>
    createMiddleware<HonoEnv>(async (c, next) => {
        // Allow all operations if the permissions include a wildcard "*"
        if (permissions.includes("*")) {
            await next();
            return;
        }

        const currentUser = c.var.currentUser;
        // Proceed if the user exists and has any of the required permissions
        if (currentUser) {
            const hasPermission = currentUser.permissions.some((p) =>
                permissions.includes(p)
            );
            if (hasPermission || permissions.includes("authenticated-only")) {
                await next();
            } else {
                unauthorized();
            }
        } else if (permissions.includes("guest-only")) {
            await next();
        } else {
            // No current user found, trigger unauthorized error
            unauthorized();
        }
    });

export default checkPermission;
```

Gambar 21. Middleware Check Permission pada Struktur Middleware

Fungsi: Digunakan untuk memeriksa apakah pengguna saat ini memiliki izin yang diperlukan. Middleware ini memeriksa apakah izin pengguna saat ini termasuk salah satu izin yang ditentukan yang diperlukan untuk melanjutkan. Ini memungkinkan untuk melanjutkan jika pengguna memiliki izin yang diperlukan atau menolak akses dengan memicu kesalahan yang tidak sah.

- Middleware staticFolder



```
const staticFolder = (path: string) =>
    serveStatic({
        root: path,
        rewriteRequestPath: (path) => path.replace(/.*\//, "./"),
    });

export default staticFolder;
```

Gambar 22. Middleware Static Folder pada Struktur Middleware

Fungsi: Digunakan untuk menyajikan file statis dari folder yang ditentukan dan menulis ulang jalur permintaan.

3. **Konfigurasi database PostgreSQL**

Berikut adalah langkah - langkah untuk pemasangan PostgreSQL:

1. Download PostgreSQL
   - Kunjungi official website PostgreSQL dan pilih versi PostgreSQL yang sesuai dengan sistem operasi perangkat Anda (Windows, macOS, atau Linux).
   - Untuk pengguna Windows, pilih Windows dan unduh installer dari EnterpriseDB. Untuk pengguna macOS dan Linux, pilih sesuai dengan instruksi di situs tersebut.
2. Instalasi PostgreSQL
   - ➢ Jalankan file installer yang telah diunduh dan ikuti panduan instalasi dari installer tersebut.
   - ➢ Saat proses instalasi, Anda akan diminta untuk memilih opsi seperti:
     - Direktori penyimpanan data (default biasanya sudah cukup).
     - Port yang digunakan (biasanya menggunakan port 5432, biarkan *default*).
     - *Password* untuk *user* 'postgres' (pilih *password* yang aman dan simpan *password* ini karena akan digunakan untuk mengakses PostgreSQL).
     - Pilih komponen tambahan (biasanya biarkan opsi default).
   - ➢ Setelah itu, lanjutkan proses instalasi dan tunggu hingga selesai.
3. Verifikasi Instalasi

   Setelah instalasi selesai, buka terminal atau command prompt, dan jalankan perintah berikut untuk memeriksa versi PostgreSQL yang terpasang:

   ```
   psql --version
   ```

Jika instalasi berhasil, perintah tersebut akan menampilkan versi PostgreSQL yang terpasang di perangkat.

4. Akses PostgreSQL
    - Untuk mengakses PostgreSQL, buka terminal atau *command prompt*, kemudian jalankan perintah berikut:
      `psql -U postgres`
      Setelah itu, Anda akan diminta untuk memasukkan password yang telah Anda terapkan pada saat instalasi.
    - Jika berhasil, Anda akan masuk ke *prompt* PostgreSQL yang dapat digunakan untuk menjalankan *query* SQL.
5. Menggunakan PostgreSQL
    - Untuk membuat database baru, Anda dapat menjalankan perintah SQL berikut di dalam PostgreSQL *prompt*:
      `CREATE DATABASE nama_database;`
    - Untuk keluar dari PostgreSQL prompt, ketikkan perintah: `\q`

4. Setup otentikasi dan autorisasi

Sistem otentikasi dan autorisasi diimplementasikan dengan menggunakan framework HonoJS dan Drizzle ORM, bersama beberapa library seperti jsonwebtoken untuk token JWT dan bcrypt untuk hashing password.

1. Definisi Data Permissions dan Roles
   File permissions.ts dan roles.ts mendefinisikan struktur permission dan role dalam aplikasi.
    - permissions.ts: Berisi daftar permissions yang digunakan untuk mengatur hak akses pengguna.

```ts
const permissionsData = [
    {
        code: "dev-routes",
    },
    {
        code: "users.readAll",
    },
] as const;

export type SpecificPermissionCode = (typeof permissionsData)[number]["code"];

export type PermissionCode =
    | SpecificPermissionCode
    | "*"
    | "authenticated-only"
    | "guest-only";

export default permissionsData;
```

Gambar 23. Permission Data pada Backend

- roles.ts: Mendefinisikan data role dengan properti:
    1. code: Identitas unik role.
    2. description: Penjelasan tentang role.
    3. permissions: Daftar kode permission yang dimiliki oleh role tersebut.

```ts
import permissionsData, { SpecificPermissionCode } from "./permissions";

export type RoleData = {
    code: RoleCode;
    description: string;
    isActive: boolean;
    name: string;
    permissions: SpecificPermissionCode[];
};

const roleData: RoleData[] = [
    {
        code: "super-admin",
        description:
            "Has full access to the system and can manage all features and settings",
        isActive: true,
        name: "Super Admin",
        permissions: permissionsData.map((permission) => permission.code),
    },
    {
        code: "user",
        description:
            "User with standard access rights for general usage of the application.",
        isActive: true,
        name: "User",
        permissions: permissionsData.map((permission) => permission.code),
    },
];

// Manually specify the union of role codes
export type RoleCode = "super-admin" | "user" | "*";

const exportedRoleData = roleData;

export default exportedRoleData;
```

Gambar 24. Role Data pada Backend

2. Skema Database

   Skema dibuat menggunakan Drizzle ORM dan mencakup tabel berikut:

- users untuk menyimpan data pengguna.

```
import { createId } from "@paralleldrive/cuid2";
import { relations } from "drizzle-orm";
import {
    boolean,
    pgTable,
    text,
    timestamp,
    varchar,
} from "drizzle-orm/pg-core";
import { permissionsToUsers } from "./permissionsToUsers";
import { rolesToUsers } from "./rolesToUsers";
import { respondents } from "./respondents";

export const users = pgTable("users", {
    id: varchar("id", { length: 50 })
        .primaryKey()
        .$defaultFn(() => createId()),
    name: varchar("name", { length: 255 }).notNull(),
    username: varchar("username").notNull().unique(),
    email: varchar("email").notNull().unique(),
    password: text("password").notNull(),
    isEnabled: boolean("isEnabled").default(true),
    resetPasswordToken: varchar("resetPasswordToken"),
    createdAt: timestamp("createdAt", { mode: "date" }).defaultNow(),
    updatedAt: timestamp("updatedAt", { mode: "date" }).defaultNow(),
    deletedAt: timestamp("deletedAt", { mode: "date" }),
});

export const usersRelations = relations(users, ({ many, one}) => ({
    permissionsToUsers: many(permissionsToUsers),
    rolesToUsers: many(rolesToUsers),
    respondent: one(respondents, {
        fields: [users.id],
        references: [respondents.userId],
    }),
}));
```

Gambar 25. Schema Users pada Backend

- roles untuk menyimpan informasi role.

```
import { createId } from "@paralleldrive/cuid2";
import { relations } from "drizzle-orm";
import { pgTable, text, timestamp, varchar } from "drizzle-orm/pg-core";
import { permissionsToRoles } from "./permissionsToRoles";

export const rolesSchema = pgTable("roles", {
    id: varchar("id", { length: 50 })
        .primaryKey()
        .$defaultFn(() => createId()),
    code: varchar("code", { length: 50 }).notNull().unique(),
    name: varchar("name", { length: 255 }).notNull(),
    description: varchar("description", { length: 255 }),
    createdAt: timestamp("createdAt").defaultNow(),
    updatedAt: timestamp("updatedAt").defaultNow(),
});

export const rolesRelations = relations(rolesSchema, ({ many }) => ({
    permissionsToRoles: many(permissionsToRoles),
}));
```

Gambar 26. Schema Roles pada Backend

- permissions untuk menyimpan daftar permissions.

```
import { createId } from "@paralleldrive/cuid2";
import { relations } from "drizzle-orm";
import { pgTable, text, timestamp, varchar } from "drizzle-orm/pg-core";
import { permissionsToUsers } from "./permissionsToUsers";
import { permissionsToRoles } from "./permissionsToRoles";

export const permissionsSchema = pgTable("permissions", {
    id: varchar("id", { length: 50 })
        .primaryKey()
        .$defaultFn(() => createId()),
    code: varchar("code", { length: 50 }).notNull().unique(),
    description: varchar("description", { length: 255 }),
    createdAt: timestamp("createdAt").defaultNow(),
    updatedAt: timestamp("updatedAt").defaultNow(),
});

export const permissionsRelations = relations(
    permissionsSchema,
    ({ many }) => ({
        permissionsToUsers: many(permissionsToUsers),
        permissionsToRoles: many(permissionsToRoles),
    })
);

```

Gambar 27. Schema Permissions pada Backend

3. Middleware authInfo

   Digunakan untuk memuat informasi otentikasi pengguna dari database. Informasi ini mencakup:

   - Query ke database untuk mendapatkan informasi pengguna, role, dan permission.
   - Menyimpan data pengguna ke dalam konteks middleware untuk digunakan di route berikutnya.

```ts
const authInfo = createMiddleware<HonoEnv>(async (c, next) => {
    const { uid, currentUser } = c.var;

    // Proceed only if uid is present and currentUser is not already set
    if (uid && !currentUser) {
        const user = await db
            .select()
            .from(users)
            .where(
                and(
                    eq(users.id, uid),
                    eq(users.isEnabled, true),
                    isNull(users.deletedAt)
                )
            )
            .leftJoin(
                permissionsToUsers,
                eq(permissionsToUsers.userId, users.id)
            )
            .leftJoin(rolesToUsers, eq(rolesToUsers.userId, users.id))
            .leftJoin(rolesSchema, eq(rolesToUsers.roleId, rolesSchema.id))
            .leftJoin(
                permissionsToRoles,
                eq(permissionsToRoles.roleId, rolesSchema.id)
            )
            .leftJoin(
                permissionsSchema,
                or(
                    eq(permissionsSchema.id, permissionsToUsers.permissionId),
                    eq(permissionsSchema.id, permissionsToRoles.permissionId)
                )
            );

        const roles = new Set<RoleCode>();
        const permissions = new Set<SpecificPermissionCode>();

        user.forEach((user) => {
            if (user.roles?.code) {
                roles.add(user.roles.code as RoleCode);
            }

            if (user.permissions?.code) {
                permissions.add(
                    user.permissions.code as SpecificPermissionCode
                );
            }
        });

        // Setting the currentUser with fetched data
        c.set("currentUser", {
            id: user[0].users.id, // Adding user ID here
            name: user[0].users.name, // Assuming the first result is the user
            permissions: Array.from(permissions),
            roles: Array.from(roles),
        });
    }

    await next();
});
```

Gambar 28. Middleware Auth Info pada Otentikasi dan Autorisasi

4. Middleware checkPermission

   Digunakan untuk memeriksa apakah pengguna memiliki permission yang diperlukan untuk mengakses route tertentu. Middleware ini dapat digunakan sebagai guard pada route.

```
import { createMiddleware } from "hono/factory";
import { PermissionCode } from "../data/permissions";
import HonoEnv from "../types/HonoEnv";
import { unauthorized } from "../errors/DashboardError";

/**
 * Creates a middleware to check if the current user has the required permissions.
 *
 * This middleware checks if the current user's permissions include any of the specified
 * permissions required to proceed. It allows proceeding if the user has the requisite
 * permissions or denies access by triggering an unauthorized error.
 *
 * @param permissions - An array of permissions to check against the current user's permissions.
 * @returns A middleware function for the Hono framework.
 */
const checkPermission = (...permissions: PermissionCode[]) =>
    createMiddleware<HonoEnv>(async (c, next) => {
        // Allow all operations if the permissions include a wildcard "*"
        if (permissions.includes("*")) {
            await next();
            return;
        }

        const currentUser = c.var.currentUser;
        // Proceed if the user exists and has any of the required permissions
        if (currentUser) {
            const hasPermission = currentUser.permissions.some((p) =>
                permissions.includes(p)
            );
            if (hasPermission || permissions.includes("authenticated-only")) {
                await next();
            } else {
                unauthorized();
            }
        } else if (permissions.includes("guest-only")) {
            await next();
        } else {
            // No current user found, trigger unauthorized error
            unauthorized();
        }
    });

export default checkPermission;

```

Gambar 29. Middleware Check Permission pada Otentikasi dan Autorisasi

5. Utilitas untuk Otentikasi

   File authUtils.ts dan passwordUtils.ts berisi fungsi-fungsi utilitas:

   a. JWT Handling:

      - generateAccessToken untuk membuat token akses.
      - verifyAccessToken untuk memvalidasi token akses.

```
import jwt from "jsonwebtoken";
import appEnv from "../appEnv";

const accessTokenSecret = appEnv.ACCESS_TOKEN_SECRET;
const refreshTokenSecret = appEnv.REFRESH_TOKEN_SECRET;
const resetPasswordTokenSecret = appEnv.RESET_PASSWORD_TOKEN_SECRET;

const algorithm: jwt.Algorithm = "HS256";

export const accessTokenExpiry: number | string | null = null;
export const refreshTokenExpiry: number | string | null = "30d";
export const resetPasswordTokenExpiry: number | string | null = null;

interface AccessTokenPayload {
    uid: string;
}

interface RefreshTokenPayload {
    uid: string;
}

interface ResetPasswordTokenPayload {
    uid: string;
}
```

Gambar 30.1 Auth Utils Variabel pada Backend

```
export const generateAccessToken = async (payload: AccessTokenPayload) => {
    const token = jwt.sign(payload, accessTokenSecret, {
        algorithm,
        ...(accessTokenExpiry ? { expiresIn: accessTokenExpiry } : {}),
    });
    return token;
};

export const generateRefreshToken = async (payload: RefreshTokenPayload) => {
    const token = jwt.sign(payload, refreshTokenSecret, {
        algorithm,
        ...(refreshTokenExpiry ? { expiresIn: refreshTokenExpiry } : {}),
    });
    return token;
};

export const verifyAccessToken = async (token: string) => {
    try {
        const payload = jwt.verify(
            token,
            accessTokenSecret
        ) as AccessTokenPayload;
        return payload;
    } catch {
        return null;
    }
};
```

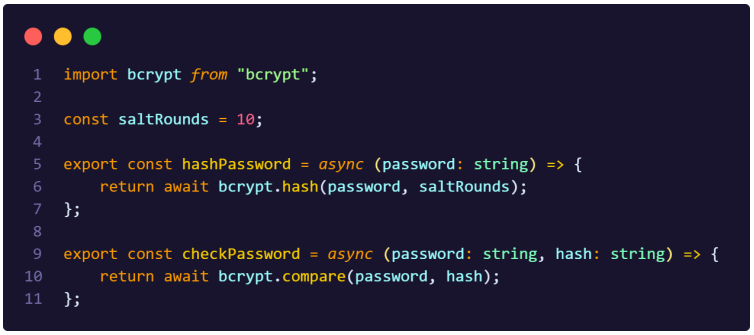
Gambar 30.2 Auth Utils Function 1 pada Backend

```
export const verifyRefreshToken = async (token: string) => {
    try {
        const payload = jwt.verify(
            token,
            refreshTokenSecret
        ) as RefreshTokenPayload;
        return payload;
    } catch {
        return null;
    }
};

export const generateResetPasswordToken = async (payload: ResetPasswordTokenPayload) => {
    const token = jwt.sign(payload, resetPasswordTokenSecret, {
        algorithm,
        ...(resetPasswordTokenExpiry ? { expiresIn: resetPasswordTokenExpiry } : {}),
    });
    return token;
};

export const verifyResetPasswordToken = async (token: string) => {
    try {
        const payload = jwt.verify(
            token,
            resetPasswordTokenSecret
        ) as ResetPasswordTokenPayload;
        return payload;
    } catch {
        return null;
    }
};
```

Gambar 30.3 Auth Utils Function 2 pada Backend

b. Password Handling:

- hashPassword untuk melakukan hashing pada password pengguna.
- checkPassword untuk memverifikasi password dengan hash.



```
import bcrypt from "bcrypt";

const saltRounds = 10;

export const hashPassword = async (password: string) => {
    return await bcrypt.hash(password, saltRounds);
};

export const checkPassword = async (password: string, hash: string) => {
    return await bcrypt.compare(password, hash);
};
```

Gambar 31. Password Utils pada Backend

6. Route Otentikasi

Route untuk login, refresh token, dan logout diimplementasikan dalam file routes/auth/route.ts.

```
.post(
        "/login",
        zValidator(
            "form",
            z.object({
                username: z.string(),
                password: z.string(),
                ___jwt: z.string().default("false"),
            })
        ),
        async (c) => {
            const formData = c.req.valid("form");

            const user = await db
                .select()
                .from(users)
                .where(
                    and(
                        isNull(users.deletedAt),
                        eq(users.isEnabled, true),
                        or(
                            eq(users.username, formData.username),
                            eq(users.email, formData.username)
                        )
                    )
                )
                .leftJoin(
                    permissionsToUsers,
                    eq(permissionsToUsers.userId, users.id)
                )
                .leftJoin(rolesToUsers, eq(rolesToUsers.userId, users.id))
                .leftJoin(rolesSchema, eq(rolesToUsers.roleId, rolesSchema.id))
                .leftJoin(
                    permissionsToRoles,
                    eq(permissionsToRoles.roleId, rolesSchema.id)
                )
                .leftJoin(
                    permissionsSchema,
                    or(
                        eq(
                            permissionsSchema.id,
                            permissionsToUsers.permissionId
                        ),
                        eq(
                            permissionsSchema.id,
                            permissionsToRoles.permissionId
                        )
                    )
                );

            if (!user.length) {
                throw new HTTPException(400, {
                    message: "Invalid username or password",
                });
            }

            const isSuccess = await checkPassword(
                formData.password,
                user[0].users.password
            );

            if (!isSuccess) {
                throw new HTTPException(400, {
                    message: "Invalid username or password",
                });
            }

            const accessToken = await generateAccessToken({
                uid: user[0].users.id,
            });

            const refreshToken = await generateRefreshToken({
                uid: user[0].users.id,
            });

            const cookieSecret = appEnv.COOKIE_SECRET;

            if (!cookieSecret)
                throw new HTTPException(500, {
                    message: "The 'COOKIE_SECRET' env var is not set",
                });

            const permissions = new Set<SpecificPermissionCode>();
            user.forEach((user) => {
                if (user.permissions?.code) {
                    permissions.add(
                        user.permissions.code as SpecificPermissionCode
                    );
                }
            });

            return c.json({
                accessToken,
                refreshToken,
                user: {
                    id: user[0].users.id,
                    name: user[0].users.name,
                    role: user[0].roles?.code,
                    permissions: Array.from(permissions),
                },
            });
        }
    )
```

Gambar 32. Route Login pada Backend

```
.post(
        "/refresh-token",
        zValidator(
            "json",
            z.object({
                refreshToken: z.string(),
            })
        ),
        async (c) => {
            const { refreshToken } = c.req.valid("json");

            const decoded = await verifyRefreshToken(refreshToken);

            if (!decoded) {
                throw new HTTPException(401, {
                    message: "Invalid refresh token",
                });
            }

            const accessToken = await generateAccessToken({
                uid: decoded.uid,
            });

            return c.json({
                accessToken,
            });
        }
    )
```

Gambar 33. Route Refresh Token pada Backend

```
.get("/logout", (c) => {
        const uid = c.var.uid;

        if (!uid) {
            return c.notFound();
        }

        deleteCookie(c, "access_token", {
            secure: true,
            httpOnly: true,
            prefix: "secure",
        });

        return c.json({
            message: "Logged out successfully",
        });
    });
```

Gambar 34. Route Logout pada Backend

5. Testing endpoint API

a. Register

Endpoint ini digunakan untuk melakukan pendaftaran akun untuk mendapatkan akses ke aplikasi

Route	/register
Method	POST
Request	JSON raw

Tabel 1. *Route Register*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
name	body	String	Berisikan nama user yang bersangkutan.
username	body	String	Berisi rangkaian huruf dan atau tanpa angka yang dibuat user sebagai identitas akun
email	body	String	Alamat email yang valid dan digunakan untuk keperluan komunikasi atau otentikasi user.
password	body	String	Berisi kata sandi yang digunakan untuk mengamankan akses ke akun user.
companyName	body	String	Nama perusahaan tempat user bekerja atau terdaftar.
position	body	String	Jabatan atau posisi user dalam perusahaan atau organisasi.

workExperience	body	String	Pengalaman kerja user yang mencakup riwayat pekerjaan sebelumnya atau saat ini.
address	body	String	Alamat tempat tinggal user, baik secara lengkap maupun hanya kota.
phoneNumber	body	String	Nomor telepon yang valid dan dapat digunakan untuk menghubungi user.

Tabel 2. *Parameter Register*

- Response

```
{
    "message": "User created successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	User created successfully
400	Bad Request	Email or username has been registered
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 3. Status Code Sign In

b. Sign In

Endpoint ini digunakan untuk masuk ke aplikasi melalui akun yang telah terdaftar sebelumnya

Route	/auth/login

Method	POST
Request	Form Data

Tabel 4. *Route Sign In*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
username	body	String	Berisi rangkaian huruf dan atau tanpa angka yang dibuat user sebagai identitas akun
password	body	String	Berisi kata sandi yang digunakan untuk mengamankan akses ke akun user.

Tabel 5. *Parameter Sign In*

- Response


```
{
"accessToken":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1aWQiOiJ3amRpNTJ
2dWt2OGFjajFrazc0dDFoYnkiLCJpYXQiOjE3MzUwNDcwODB9.UyQHpI-
MJV4BvgKJFZ1FcPQA50fZNGb9ZVBp46TPxjI",
"refreshToken":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1aWQiOiJ3amRpNTJ
2dWt2OGFjajFrazc0dDFoYnkiLCJpYXQiOjE3MzUwNDcwODAsImV4cCI6
MTczNzYzOTA4MH0.YkXE_kKJJdXmbnrHYAJ3k5ff4w8DLI_s6pRHH6iiW
R0",
    "user": {
        "id": "wjdi52vukv8acj1kk74t1hby",
        "name": "respondenUji01",
        "role": "user",
        "permissions": [
            "dev-routes",
            "users.readAll",
            "users.create",
```

```
            "users.update",
            "users.delete",
            "users.restore",
            "permissions.read",
            "roles.read",
            "roles.create",
            "roles.update",
            "roles.delete",
            "questions.readAll",
            "questions.create",
            "questions.update",
            "questions.delete",
            "questions.restore",
            "assessmentRequestManagement.readAll",
            "assessmentRequestManagement.update",
            "assessmentRequestManagement.read",
            "managementAspect.readAll",
            "managementAspect.create",
            "managementAspect.update",
            "managementAspect.delete",
            "managementAspect.restore",
            "assessmentResult.readAll",
            "assessmentResult.read",
            "assessmentResult.readAllQuestions",
            "assessmentResult.create",
            "assessmentRequest.read",
            "assessmentRequest.create",
            "assessmentRequest.update",
            "assessments.readAspect",
            "assessments.readAllQuestions",
            "assessments.readAnswers",
            "assessments.toggleFlag",
            "assessments.uploadFile",
            "assessments.submitOption",
            "assessments.submitValidation",
            "assessments.submitAssessment",
            "assessments.readAverageSubAspect",
            "assessments.readAverageAspect",
            "assessments.updateOption",
            "assessmentResult.update"
        ]
    }
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Login successfully
400	Bad Request	Invalid username or password

Tabel 6. *Status Code Sign In*

c. Refresh Token

Endpoint ini digunakan untuk memperbarui token akses yang telah didapat ketika melakukan proses sign in sebelumnya.

Route	/auth/refresh-token
Method	POST
Request	JSON raw

Tabel 7. *Route Refresh Token*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
refreshToken	body	String	Token autentikasi unik yang dihasilkan saat user berhasil login. Digunakan untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 8. *Parameter Refresh Token*

- Response

```
{
```

```
"accessToken":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1aWQiOiJ3amRpNTJ
2dWt2OGFjajFrazc0dDFoYnkiLCJpYXQiOjE3MzUwNDg0MTd9.6yaS_QT
sdgwBx2GdTbhQjEsifQ-PO7GB6RerMU3zFFc"
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Success
401	Unauthorized	Invalid refresh token

Tabel 9. *Status Code Refresh Token*

d. Sign Out

Endpoint ini digunakan untuk keluar dari aplikasi

Route	/auth/logout
Method	GET
Request	Bearer Token

Tabel 10. *Route Sign Out*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login. Digunakan untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 11. *Parameter Sign Out*

- Response

```
{
    "message": "Logged out successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Logged out successfully
404	Not Found	Not Found

Tabel 12. *Status Code Sign Out*

e. Forgot Password

Endpoint ini digunakan ketika user melupakan kata sandi yang terdaftar untuk akses masuk ke aplikasi.

Route	/forgot-password
Method	POST
Request	JSON raw

Tabel 13. *Route Forgot Password*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
email	body	String	Alamat email yang valid dan digunakan untuk keperluan komunikasi atau otentikasi user.

Tabel 14. *Parameter Forgot Password*

- Response

```
{
    "message": "Email has been sent successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Logged out successfully
400	Bad Request	Invalid email

Tabel 15. *Status Code Forgot Password*

f. Reset Password

Endpoint ini digunakan ketika user ingin mengatur ulang kata sandi akun yang terdaftar di aplikasi

Route	/forgot-password/verify?token=
Method	PATCH
Request	- Query Params - JSON raw

Tabel 16. *Route Reset Password*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
token	param	String	Token autentikasi unik yang dikirimkan melalui email untuk

			kunci akses melakukan reset password.
password	body	String	Berisi kata sandi yang digunakan untuk mengamankan akses ke akun user.
confirm_password	body	String	Berisi kata sandi yang sama dengan kata sandi yang diinputkan sebelumnya

Tabel 17. *Parameter Reset Password*

- Response

```
{
    "message": "Password has been reset successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Logged out successfully
401	Unauthorized	Invalid or expired token

Tabel 18. *Status Code Reset Password*

g. Users List

Endpoint ini digunakan untuk menampilkan seluruh daftar dan detail data dari pengguna yang terdaftar di aplikasi.

Route	/users
Method	GET
Request	Bearer Token

Tabel 19. *Route Users List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 20. *Parameter Users List*

- Response


```
{
    "data": [
        {
            "id": "pjbxwhpigrofs1cymc96oxdg",
            "name": "Super Admin",
            "email": "admin@admin.com",
            "username": "superadmin",
            "isEnabled": true,
            "createdAt": "2024-12-11T14:10:22.964Z",
            "updatedAt": "2024-12-11T14:10:22.964Z",
            "company": null,
            "roles": [
                {
                    "id": "tar402g95qu31ou9t9zs5hbt",
                    "name": "Super Admin"
                }
            ]
        },
        {
            "id": "wjdi52vukv8acj1kk74t1hby",
            "name": "respondenUji01",
            "email": "figara4470@kelenson.com",
            "username": "uji01",
            "isEnabled": true,
            "createdAt": "2024-12-24T20:07:16.523Z",
            "updatedAt": "2024-12-24T14:04:45.802Z",
```

```
            "company": "konoha",
            "roles": [
                {
                    "id": "spqfxlu7l766zqbsfu86xv9e",
                    "name": "User"
                }
            ]
        }
    ],
    "_metadata": {
        "currentPage": 0,
        "totalPages": 1,
        "totalItems": "2",
        "perPage": 10
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Users list
401	Unauthorized	Unauthorized

Tabel 21. *Status Code Users List*

h. User by ID

Endpoint ini digunakan untuk menampilkan data dari pengguna tertentu menggunakan ID pengguna.

Route	/users/{id}
Method	GET
Request	- Bearer Token

Tabel 22. *Route User*

- Parameters

Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 23. *Parameter User*

- Response


```
{
    "id": "wjdi52vukv8acj1kk74t1hby",
    "name": "respondenUji01",
    "position": "anbu",
    "workExperience": "2 years",
    "email": "figara4470@kelenson.com",
    "companyName": "konoha",
    "address": "hinokuni",
    "phoneNumber": "081234567890",
    "username": "uji01",
    "isEnabled": true,
    "createdAt": "2024-12-24T20:07:16.523Z",
    "updatedAt": "2024-12-24T14:04:45.802Z",
    "roles": [
        {
            "id": "spqfxlu7l766zqbsfu86xv9e",
            "name": "User"
        }
    ]
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message

200	Ok	User data
401	Unauthorized	Unauthorized

Tabel 24. *Status Code User*

i. Create User

Endpoint ini digunakan untuk menambahkan data pengguna baru ke dalam aplikasi yang hanya bisa dilakukan oleh *superadmin.*

Route	/users
Method	POST
Request	- Bearer Token - JSON raw

Tabel 25. *Route Create User*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
name	body	String	Berisikan nama user yang bersangkutan.
username	body	String	Berisi rangkaian huruf dan atau tanpa angka yang dibuat user sebagai identitas akun
email	body	String	Alamat email yang valid dan

			digunakan untuk keperluan komunikasi atau otentikasi user.
password	body	String	Berisi kata sandi yang digunakan untuk mengamankan akses ke akun user.
companyName	body	String	Nama perusahaan tempat user bekerja atau terdaftar.
position	body	String	Jabatan atau posisi user dalam perusahaan atau organisasi.
workExperience	body	String	Pengalaman kerja user yang mencakup riwayat pekerjaan sebelumnya atau saat ini.
address	body	String	Alamat tempat tinggal user, baik secara lengkap maupun hanya kota.
phoneNumber	body	String	Nomor telepon yang valid dan dapat digunakan untuk menghubungi user.
roles	body	Array of string	Berisi ID dari *roles* yang digunakan untuk menentukan hak akses atau izin tertentu dalam sistem.

Tabel 26. *Parameter Create User*

- Response

```
{
    "message": "User created successfully"
}
```

- Errors

Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
201	Created	User created successfully
401	Unauthorized	Unauthorized
403	Forbidden	Email or username has been registered
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 27. *Status Code Create User*

j. Update User

Endpoint ini digunakan untuk melakukan pembaruan data pengguna tertentu menggunakan ID pengguna terkait yang ingin diubah oleh *superadmin.*

Route	/users/{id}
Method	PATCH
Request	- Bearer Token - JSON raw

Tabel 28. *Route Update User*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil

			login untuk mengakses endpoint yang membutuhkan otorisasi.
name	body	String	Berisikan nama user yang bersangkutan.
username	body	String	Berisi rangkaian huruf dan atau tanpa angka yang dibuat user sebagai identitas akun
email	body	String	Alamat email yang valid dan digunakan untuk keperluan komunikasi atau otentikasi user.
password	body	String	Berisi kata sandi yang digunakan untuk mengamankan akses ke akun user.
companyName	body	String	Nama perusahaan tempat user bekerja atau terdaftar.
position	body	String	Jabatan atau posisi user dalam perusahaan atau organisasi.
workExperience	body	String	Pengalaman kerja user yang mencakup riwayat pekerjaan sebelumnya atau saat ini.
address	body	String	Alamat tempat tinggal user, baik secara lengkap maupun hanya kota.
phoneNumber	body	String	Nomor telepon yang valid dan dapat digunakan untuk menghubungi user.

roles	body	Array of string	Berisi ID dari *roles* yang digunakan untuk menentukan hak akses atau izin tertentu dalam sistem.

Tabel 29. *Parameter Update User*

- Response

```
{
    "message": "User updated successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	User updated successfully
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 30. *Status Code Update User*

k. Delete User

Endpoint ini digunakan untuk menghapus data pengguna yang hanya bisa dilakukan oleh *superadmin.*

Route	/users/{id}
Method	DELETE
Request	Bearer Token

Tabel 31. *Route Delete User*

- Parameters

Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 32. *Parameter Delete User*

- Response

```
{
    "message": "User deleted successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	User deleted successfully
401	Unauthorized	Unauthorized

Tabel 33. *Status Code Delete User*

l. Undo Delete User

Endpoint ini digunakan untuk mengembalikan data pengguna yang telah terhapus dan hanya bisa dilakukan oleh *superadmin.*

Route	/users/restore/{id}
Method	PATCH
Request	Bearer Token

Tabel 34. *Route Undo Delete User*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 35. *Parameter Undo Delete User*

- Response

```
{
    "message": "User restored successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	User restored successfully
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 36. *Status Code Undo Delete User*

m. Roles List

Endpoint ini digunakan untuk menampilkan daftar *roles* dalam aplikasi.

Route	/roles

Method	GET
Request	Bearer Token

Tabel 37. *Route Roles List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 38. *Parameter Roles List*

- Response

```
[
    {
        "id": "tar402g95qu31ou9t9zs5hbt",
        "code": "super-admin",
        "name": "Super Admin"
    },
    {
        "id": "spqfxlu7l766zqbsfu86xv9e",
        "code": "user",
        "name": "User"
    }
]
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message

200	Ok	Roles List Data
401	Unauthorized	Unauthorized

Tabel 39. *Status Code Roles List*

n. Permissions List

Endpoint ini digunakan untuk menampilkan daftar *permission* atau ijin akses dalam aplikasi.

Route	/permissions
Method	GET
Request	Bearer Token

Tabel 40. *Route Permissions List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 42. *Parameter Permissions List*

- Response

```
[
    {
        "id": "soa0ud4dztfxinitqn1jq4n1",
        "code": "dev-routes"
    },
    . . .,
    {
        "id": "qjz52oqkunji8uekkq00yga3",
```

```
        "code": "assessmentResult.update"
    }
]
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Permissions List Data
401	Unauthorized	Unauthorized

Tabel 43. *Status Code Permissions List*

o. Aspects and Sub Aspects List

Endpoint ini digunakan untuk menampilkan daftar aspek dan sub aspek dalam aplikasi.

Route	/management-aspect
Method	GET
Request	Bearer Token

Tabel 44. *Route Aspects and Sub Aspects List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 45. *Parameter Aspects and Sub Aspects List*

- Response


```
{
    "data": [
        {
            "id": "gsgk0aic5n4ef8lbyq8covli",
            "name": "Deteksi",
            "createdAt": "2024-12-11T07:10:23.034Z",
            "updatedAt": "2024-12-11T07:10:23.034Z",
            "subAspects": [
                {
                    "id": "zli3ba1xl5ajdl0dt6rgljgr",
                     "name": "Mengelola deteksi Peristiwa
Siber",
                    "questionCount": "7"
                },
                {
                    "id": "hn58dzqkdb42clnlv0khlm42",
                        "name": "Memantau Peristiwa Siber
berkelanjutan",
                    "questionCount": "7"
                },
                {
                    "id": "jz9cfwbudje5q1zddqctmyyz",
                        "name": "Menganalisis anomali dan
Peristiwa Siber",
                    "questionCount": "5"
                }
            ]
        },
        {
            "id": "t6vn45nmrvv29ezpplg1jk6d",
            "name": "Identifikasi",
            "createdAt": "2024-12-11T07:10:23.034Z",
            "updatedAt": "2024-12-11T07:10:23.034Z",
            "subAspects": [
                {
                    "id": "ib08n5ahgrg72mzrtyp725vb",
                    "name": "Mengelola aset informasi",
                    "questionCount": "8"
                },
                {
                    "id": "xw3bazdspo5c249pjisgli4f",
```

```
                        "name": "Mengelola risiko rantai
pasok",
                    "questionCount": "12"
                },
                {
                    "id": "xotxn9r85fz3wka0st7xnodg",
                      "name": "Mengidentifikasi Peran dan
tanggung jawab organisasi",
                    "questionCount": "5"
                },
                {
                    "id": "y6n8hb2jkj9c982c7uymelok",
                              "name": "Menyusun strategi,
kebijakan, dan prosedur Pelindungan IIV",
                    "questionCount": "7"
                },
                {
                    "id": "gz10yc15ec95ioc83d88aft4",
                     "name": "Menilai dan mengelola risiko
Keamanan Siber",
                    "questionCount": "19"
                }
            ]
        },
        {
            "id": "p3n3sfyhncza9gcgkwc7gmaq",
            "name": "Penanggulangan dan Pemulihan",
            "createdAt": "2024-12-11T07:10:23.034Z",
            "updatedAt": "2024-12-11T07:10:23.034Z",
            "subAspects": [
                {
                    "id": "q3o1wwp92by25xg3pb387nio",
                           "name": "Meningkatkan keamanan
setelah terjadinya Insiden Siber",
                    "questionCount": "7"
                },
                {
                    "id": "w687hi0ezuswyyuivqka2rjc",
                            "name": "Menyusun perencanaan
penanggulangan dan pemulihan Insiden Siber",
                    "questionCount": "14"
                },
                {
                    "id": "mbygz73jok9hgtsjq2ofounk",
```

```
                    "name": "Menganalisis dan melaporkan
Insiden Siber",
                    "questionCount": "6"
                },
                {
                    "id": "agxhz2oho3n2lv4t4dlfi3s8",
                    "name": "Melaksanakan penanggulangan
dan pemulihan Insiden Siber",
                    "questionCount": "22"
                }
            ]
        },
        {
            "id": "yww407bnba9m7ec9njyjbzxo",
            "name": "Proteksi",
            "createdAt": "2024-12-11T07:10:23.034Z",
            "updatedAt": "2024-12-11T07:10:23.034Z",
            "subAspects": [
                {
                    "id": "lsdk38d6e2v1hz6x1t4yjtio",
                    "name": "Melindungi data",
                    "questionCount": "15"
                },
                {
                    "id": "z885tsb3rb919z1gy8ryihsf",
                    "name": "Melindungi aset fisik",
                    "questionCount": "9"
                },
                {
                    "id": "uu7kltn68ap5r5ae7yqk3mop",
                            "name": "Mengelola identitas,
autentikasi, dan kendali akses",
                    "questionCount": "6"
                },
                {
                    "id": "guusrqoipg99wd6dkhrrrw9u",
                    "name": "Melindungi jaringan",
                    "questionCount": "13"
                },
                {
                    "id": "b0t8z03qxi6uct0t2zd1y59k",
                         "name": "Melindungi sumber daya
manusia",
                    "questionCount": "13"
```

```
                },
                {
                    "id": "kruez3nrucxok4lhsofmct1q",
                    "name": "Melindungi aplikasi",
                    "questionCount": "6"
                }
            ]
        }
    ],
    "_metadata": {
        "currentPage": 0,
        "totalPages": 1,
        "totalItems": 4,
        "perPage": 10
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Aspects and Sub Aspects List Data
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 46. *Status Code Aspects and Sub Aspects List*

p. Aspect and Sub Aspect by ID

Endpoint ini digunakan untuk menampilkan aspek dan sub aspek tertentu berdasarkan ID yang dicari.

Route	/management-aspect/{id}
Method	GET
Request	Bearer Token

Tabel 47. *Route Aspect and Sub Aspect by ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 45. *Parameter Aspects and Sub Aspects List*

- Response


```
{
    "id": "gsgk0aic5n4ef8lbyq8covli",
    "name": "Deteksi",
    "createdAt": "2024-12-11T07:10:23.034Z",
    "updatedAt": "2024-12-11T07:10:23.034Z",
    "subAspects": [
        {
            "id": "hn58dzqkdb42clnlv0khlm42",
                    "name": "Memantau Peristiwa Siber
berkelanjutan",
            "questionCount": 7
        },
        {
            "id": "jz9cfwbudje5q1zddqctmyyz",
              "name": "Menganalisis anomali dan Peristiwa
Siber",
            "questionCount": 5
        },
        {
            "id": "zli3ba1xl5ajdl0dt6rgljgr",
            "name": "Mengelola deteksi Peristiwa Siber",
            "questionCount": 7
        }
    ]
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Aspects and Sub Aspects Data
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 48. *Status Code Aspect and Sub Aspect by ID*

q. Create Aspect and Sub Aspect

Endpoint ini digunakan untuk menambah data aspek dan sub aspek baru ke dalam aplikasi.

Route	/management-aspect
Method	POST
Request	- Bearer Token - JSON raw

Tabel 49. *Route Create Aspect and Sub Aspect*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

name	body	String	Nama dari aspek yang ingin dibuat
subAspects	body	Array of String	Daftar nama sub aspek yang ingin dibuat dan terkait dengan aspek sebelumnya

Tabel 50. *Parameter Create Aspect and Sub Aspect*

- Response

```
{
        "message": "Aspect and sub aspects created successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
201	Created	Aspect and Sub Aspect created successfully
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 51. *Status Code Create Aspect and Sub Aspect*

r. Update Aspect and Sub Aspect

Endpoint ini digunakan untuk memperbarui aspek dan sub aspek tertentu berdasarkan ID yang dicari.

Route	/management-aspect/{id}
Method	PATCH
Request	- Bearer Token

	- JSON raw

Tabel 52. *Route Update Aspect and Sub Aspect*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
name	body	String	Nama dari aspek yang ingin dibuat
subAspects	body	Array of String	Daftar sub aspek yang terkait dengan aspek sebelumnya, dimana terdapat variabel didalamnya
id	body	String	ID milik tiap sub aspek yang terkait dengan aspek
name	body	String	Nama milik tiap sub aspek yang terkait dengan aspek
aspectId	body	String	ID milik aspek yang diperbarui datanya

Tabel 53. *Parameter Update Aspect and Sub Aspect*

- Response

```
{
```

```
        "message": "Aspect and sub aspects updated
successfully"
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Aspect and Sub Aspect updated successfully
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 54. *Status Code Update Aspect and Sub Aspect*

s. Delete Aspect and Sub Aspect

Endpoint ini digunakan untuk menghapus aspek dan sub aspek tertentu berdasarkan ID yang dicari.

Route	/management-aspect/{id}
Method	DELETE
Request	Bearer Token

Tabel 55. *Route Delete Aspect and Sub Aspect*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin

			berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 56. *Parameter Delete Aspect and Sub Aspect*

- Response

```
{
      "message": "Aspect and sub aspects soft deleted
successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Aspect and Sub Aspect soft deleted successfully
401	Unauthorized	Unauthorized

Tabel 57. *Status Code Delete Aspect and Sub Aspect*

t. Undo Delete Aspect and Sub Aspect

Endpoint ini digunakan untuk mengembalikan data aspek dan sub aspek tertentu yang telah terhapus berdasarkan ID yang dicari.

Route	/management-aspect/restore/{id}
Method	PATCH
Request	Bearer Token

Tabel 58. *Route Undo Delete Aspect and Sub Aspect*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 59. *Parameter Undo Delete Aspect and Sub Aspect*

- Response

```
{
        "message": "Aspect and sub aspects restored
successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Aspect and Sub Aspect restored successfully
401	Unauthorized	Unauthorized

Tabel 60. *Status Code Undo Delete Aspect and Sub Aspect*

u. Aspect List

Endpoint ini digunakan untuk menampilkan daftar aspek yang ada di aplikasi.

Route	/questions/aspects
Method	GET
Request	Bearer Token

Tabel 61. *Route Aspects List*

- Parameters

Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 62. *Parameter Aspects List*

- Response


```
[
    {
        "id": "t6vn45nmrvv29ezpplg1jk6d",
        "name": "Identifikasi",
        "createdAt": "2024-12-11T07:10:23.034Z",
        "updatedAt": "2024-12-11T07:10:23.034Z"
    },
    {
        "id": "yww407bnba9m7ec9njyjbzxo",
        "name": "Proteksi",
        "createdAt": "2024-12-11T07:10:23.034Z",
        "updatedAt": "2024-12-11T07:10:23.034Z"
    },
    {
        "id": "gsgk0aic5n4ef81byq8covli",
        "name": "Deteksi",
        "createdAt": "2024-12-11T07:10:23.034Z",
        "updatedAt": "2024-12-11T07:10:23.034Z"
    },
    {
        "id": "p3n3sfyhncza9gcgkwc7gmaq",
        "name": "Penanggulangan dan Pemulihan",
        "createdAt": "2024-12-11T07:10:23.034Z",
        "updatedAt": "2024-12-11T07:10:23.034Z"
    },
    {
        "id": "hc3vmiebeklremmhmr8qb3a2",
        "name": "aspek001",
```

```
        "createdAt": "2024-12-25T20:03:15.014Z",
        "updatedAt": "2024-12-25T13:08:19.351Z"
    }
]
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Aspects List Data
401	Unauthorized	Unauthorized

Tabel 63. *Status Code Aspects List*

v. Sub Aspect List

Endpoint ini digunakan untuk menampilkan daftar sub aspek yang ada di aplikasi.

Route	/questions/subAspects
Method	GET
Request	Bearer Token

Tabel 64. *Route Sub Aspects List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan

			otorisasi.

Tabel 65. *Parameter Sub Aspects List*

- Response


```
[
    {
        "id": "xotxn9r85fz3wka0st7xnodg",
           "name": "Mengidentifikasi Peran dan tanggung
jawab organisasi",
        "aspectId": "t6vn45nmrvv29ezpplg1jk6d",
        "createdAt": "2024-12-11T07:10:23.059Z",
        "updatedAt": "2024-12-11T07:10:23.059Z"
    },
       . . .,
    {
        "id": "rbsv8sx0c2qkvjugozmuesqx",
        "name": "subaspek003",
        "aspectId": "hc3vmiebeklremmhmr8qb3a2",
        "createdAt": "2024-12-25T20:03:15.041Z",
        "updatedAt": "2024-12-25T13:08:19.360Z"
    }
]
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Sub Aspect List Data
401	Unauthorized	Unauthorized

Tabel 66. *Status Code Sub Aspects List*

w. Questions List

Endpoint ini digunakan untuk menampilkan daftar pertanyaan yang ada di aplikasi.

Route	/questions

Method	GET
Request	Bearer Token

Tabel 67. *Route Questions List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 68. *Parameter Questions List*

- Response


```
{
    "data": [
        {
            "id": "dol5ju6f8pguj4d8b712zr0g",
                "question": "Berbagi informasi mengenai efektivitas teknologi perlindungan data hanya dengan mitra yang tepat dan terpercaya.",
            "needFile": false,
            "aspectName": "Proteksi",
            "subAspectName": "Melindungi jaringan",
            "createdAt": "2024-12-11T14:10:23.533Z",
            "updatedAt": "2024-12-11T14:10:23.533Z",
            "averageScore": "2.50"
        },
        . . .,
        {
            "id": "q9auw1p9nnhtjg0i4l1tk8pu",
                "question": "Melakukan proses pemantauan peristiwa keamanan siber, sesuai dengan peraturan,
```

```
arahan, standar industri, dan aturan lainnya yang
berlaku.",
            "needFile": false,
            "aspectName": "Deteksi",
            "subAspectName": "Mengelola deteksi Peristiwa
Siber",
            "createdAt": "2024-12-11T14:10:23.594Z",
            "updatedAt": "2024-12-11T14:10:23.594Z",
            "averageScore": "2.50"
        }
    ],
    "_metadata": {
        "currentPage": 0,
        "totalPages": 7,
        "totalItems": 181,
        "perPage": 30
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Questions List Data
401	Unauthorized	Unauthorized

Tabel 69. *Status Code Questions List*

x. Questions by ID

Endpoint ini digunakan untuk menampilkan pertanyaan berdasarkan ID yang dicari.

Route	/questions/{id}
Method	GET
Request	Bearer Token

Tabel 70. *Route Question by ID*

- Parameters

Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 71. *Parameter Question by ID*

- Response


```
{
    "id": "dol5ju6f8pguj4d8b712zr0g",
     "question": "Berbagi informasi mengenai efektivitas teknologi perlindungan data hanya dengan mitra yang tepat dan terpercaya.",
    "needFile": false,
    "subAspectId": "guusrqoipg99wd6dkhrrrw9u",
    "subAspectName": "Melindungi jaringan",
    "aspectId": "yww407bnba9m7ec9njyjbzxo",
    "aspectName": "Proteksi",
    "createdAt": "2024-12-11T14:10:23.533Z",
    "updatedAt": "2024-12-11T14:10:23.533Z",
    "options": [
        {
            "id": "ncb4xhoghg15ujv3ajzqtkk0",
             "text": "Organisasi belum melakukan berbagi informasi mengenai efektivitas teknologi perlindungan data",
            "score": 0
        },
        {
            "id": "tzbaeik1u84164wn1c1ki1ok",
             "text": "Organisasi telah menjalankan berbagi informasi mengenai efektivitas teknologi perlindungan data hanya dengan mitra yang tepat dan terpercaya namun belum secara menyeluruh (masih sebagian kecil) baik terhadap teknologi maupun terhadap mitra, tetapi prosedurnya belum diformalkan",
```

```
            "score": 1
        },
        {
            "id": "kpleelbcpwlyw5ddrj7lisz6",
            "text": "Organisasi telah menjalankan berbagi informasi mengenai efektivitas teknologi perlindungan data hanya dengan sebagian besar mitra yang tepat dan terpercaya baik terhadap teknologi maupun terhadap mitra, tetapi prosedurnya belum diformalkan",
            "score": 2
        },
        {
            "id": "jrrbnr2ysvmacr3ryuxri4te",
            "text": "Organisasi telah menjalankan berbagi informasi mengenai efektivitas teknologi perlindungan data hanya dengan mitra yang tepat dan terpercaya secara menyeluruh baik terhadap teknologi maupun terhadap mitra, dan prosedurnya sudah diformalkan",
            "score": 3
        },
        {
            "id": "knfh1icvubeqqeuwiezke4mp",
            "text": "Organisasi telah menjalankan berbagi informasi mengenai efektivitas teknologi perlindungan data hanya dengan mitra yang tepat dan terpercaya secara menyeluruh baik terhadap teknologi maupun terhadap mitra, dan prosedurnya sudah diformalkan, dimonitoring, dan direview secara berkala",
            "score": 4
        },
        {
            "id": "idk17owim1yvork1lb5htxoy",
            "text": "Organisasi telah menjalankan berbagi informasi mengenai efektivitas teknologi perlindungan data hanya dengan mitra yang tepat dan terpercaya secara menyeluruh baik terhadap teknologi maupun terhadap mitra, dan prosedurnya sudah diformalkan, dimonitoring, direview secara berkala, dan dilakukan perbaikan",
            "score": 5
        }
    ]
}
```


- Errors

Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Question Data by ID
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 72. *Status Code Question by ID*

y. Create Question

Endpoint ini digunakan untuk menambahkan pertanyaan ke dalam aplikasi.

Route	/questions
Method	POST
Request	- Bearer Token - JSON raw

Tabel 73. *Route Create Question*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
question	body	String	Kalimat pertanyaan yang ingin

			dimasukkan ke dalam aplikasi
subAspectId	body	String	ID milik sub aspek yang berkaitan dengan pertanyaan yang ingin dimasukkan
needFile	body	Boolean	Keterangan kebutuhan dokumen pendukung jawaban dari pertanyaan terkait
options	body	Array of object	Daftar pilihan jawaban untuk pertanyaan terkait dimana terdapat “text” dan “score” dalam setiap *options*
text	body	String	Kalimat berupa jawaban yang terkait dengan pertanyaan
score	body	Integer	Nilai dari jawaban yang dimiliki oleh tiap pilihan jawaban

Tabel 74. *Parameter Create Question*

- Response


```
{
          "message": "Question and options created
successfully",
    "data": {
        "id": "ef3in4mz83ookrb9687cppf4",
        "subAspectId": "xotxn9r85fz3wka0st7xnodg",
        "question": "Uji Coba API CREATE QUESTION",
        "needFile": false,
        "createdAt": "2024-12-27T08:25:36.071Z",
        "updatedAt": "2024-12-27T08:25:36.071Z",
        "deletedAt": null
    }
}
```


- Errors

Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
201	Created	Question and options created successfully
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 75. *Status Code Create Question*

z. Update Question

Endpoint ini digunakan untuk mengedit data pertanyaan ke dalam aplikasi.

Route	/questions/{id}
Method	PATCH
Request	- Bearer Token - JSON raw

Tabel 76. *Route Update Question*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

question	body	String	Kalimat pertanyaan yang ingin dimasukkan ke dalam aplikasi
subAspectId	body	String	ID milik sub aspek yang berkaitan dengan pertanyaan yang ingin dimasukkan
needFile	body	Boolean	Keterangan kebutuhan dokumen pendukung jawaban dari pertanyaan terkait
options	body	Array of object	Daftar pilihan jawaban untuk pertanyaan terkait dimana terdapat "text" dan "score" dalam setiap *options*
text	body	String	Kalimat berupa jawaban yang terkait dengan pertanyaan
score	body	Integer	Nilai dari jawaban yang dimiliki oleh tiap pilihan jawaban

Tabel 77. *Parameter Update Question*

- Response

```
{
        "message": "Question and options updated successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Question and options updated successfully

401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 78. *Status Code Update Question*

aa. Delete Question

Endpoint ini digunakan untuk menghapus data pertanyaan ke dalam aplikasi.

Route	/questions/{id}
Method	DELETE
Request	Bearer Token

Tabel 79. *Route Delete Question*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 80. *Parameter Delete Question*

- Response

```
{
    "message": "Question deleted successfully"
}
```

- Errors

Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Question deleted successfully
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 81. *Status Code Delete Question*

bb. Undo Delete Question

Endpoint ini digunakan untuk mengembalikan data pertanyaan yang telah terhapus ke dalam aplikasi.

Route	/questions/restore/{id}
Method	PATCH
Request	Bearer Token

Tabel 82. *Route Undo Delete Question*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 83. *Parameter Undo Delete Question*

- Response

```
{
    "message": "Question restored successfully"
```

```
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Question restored successfully
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 84. *Status Code Undo Delete Question*

cc. Assessment Request List

Endpoint ini digunakan untuk menampilkan daftar permohonan asesmen yang dimiliki tiap pengguna.

Route	/assessmentRequest
Method	GET
Request	Bearer Token

Tabel 85. *Route Assessment Request List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 86. *Parameter Assessment Request List*

- Response


```
{
    "data": [
        {
            "userId": "rk7x4y1454rr72zx7w4a6bne",
            "name": "asdfghjkl",
            "assessmentId": "ylyqvgrgc5utnml3y82jedd7",
            "tanggal": "2024-12-11T07:22:59.234Z",
            "status": "belum diverifikasi",
            "respondentId": "kld83ruwf5mne39hol45p23q"
        }
    ],
    "_metadata": {
        "currentPage": 0,
        "totalPages": 1,
        "totalItems": "1",
        "perPage": 10
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Assessment requests List Data
401	Unauthorized	Unauthorized

Tabel 87. *Status Code Assessment Request List*

dd. Create Assessment Request

Endpoint ini digunakan untuk membuat permohonan asesmen oleh pengguna.

Route	/assessmentRequest
Method	POST
Request	- Bearer Token

	- JSON raw

Tabel 88. *Route Create Assessment Request*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
respondentId	body	String	ID responden yang dimiliki oleh setiap user yang ingin melakukan permohonan asesmen

Tabel 89. *Parameter Create Assessment Request*

- Response

```
{
      "message": "Successfully submitted the assessment
request",
    "data": [
        {
            "id": "x2dh925rsz1jmbh2skcclm9b",
            "respondentId": "k1d83ruwf5mne39hol45p23q",
            "status": "menunggu konfirmasi",
            "reviewedBy": null,
            "reviewedAt": null,
            "verifiedBy": null,
            "verifiedAt": null,
            "createdAt": "2024-12-27T02:03:42.639Z"
        }
    ]
}
```

- Errors

Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
201	Created	Successfully submitted the assessment request
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 90. *Status Code Create Assessment Request*

ee. Update Assessment Request

Endpoint ini digunakan untuk mengubah status permohonan asesmen pengguna oleh *superadmin*.

Route	/assessmentRequest/{assessmentId}
Method	PATCH
Request	- Bearer Token - JSON raw

Tabel 91. *Route Update Assessment Request*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

status	body	String	Status yang dimiliki oleh data permohonan asesmen

Tabel 92. *Parameter Update Assessment Request*

- Response


```
{
        "message": "Assessment status updated started
successfully",
    "data": {
        "updatedAssessment": [
            {
                "id": "x2dh925rsz1jmbh2skcclm9b",
                                            "respondentId":
"kld83ruwf5mne39hol45p23q",
                "status": "diterima",
                "reviewedBy": null,
                "reviewedAt": null,
                "verifiedBy": null,
                "verifiedAt": null,
                "createdAt": "2024-12-27T02:03:42.639Z"
            }
        ],
        "answerRecords": [
            {
                "id": "ocn1pm71x6m7z8o7i49peqjr",
                "questionId": "qrb7anxpxdu27e0ybd01idvj",
                "optionId": null,
                "validationInformation": "",
                                            "assessmentId":
"x2dh925rsz1jmbh2skcclm9b"
            },
              . . .,
            {
                "id": "grhkoycsxtsz0a2l5puibswt",
                "questionId": "ruuxnewp4wp8rx5mfp5e0fqz",
                "optionId": null,
                "validationInformation": "",
                                            "assessmentId":
"x2dh925rsz1jmbh2skcclm9b"
            }
        ]
    }
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Assessment status updated successfully
401	Unauthorized	Unauthorized

Tabel 93. *Status Code Update Assessment Request*

ff. Assessment Request Management List

Endpoint ini digunakan untuk menampilkan daftar permohonan asesmen yang ada di aplikasi.

Route	/assessmentRequestManagement
Method	GET
Request	Bearer Token

Tabel 94 . *Route Assessment Request Management List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 95. *Parameter Assessment Request Management List*

- Response


```
"data": [
        {
            "idPermohonan": "tjqvgev1e6pohj11ll5hq2n7",
            "namaResponden": "Respondent User 3",
            "namaPerusahaan": "Company DD",
            "status": "menunggu konfirmasi",
            "tanggal": "2024-11-13T02:33:33.445Z"
        },
        {
            "idPermohonan": "x3o0ftzef22ebzswwn7vq6xi",
            "namaResponden": "Respondent User 2",
            "namaPerusahaan": "Company C",
            "status": "menunggu konfirmasi",
            "tanggal": "2024-11-13T02:33:33.445Z"
        },
        {
            "idPermohonan": "trzx2u42cwkmz5bmzz8raw5e",
            "namaResponden": "Respondent User 4",
            "namaPerusahaan": "Company FS",
            "status": "menunggu konfirmasi",
            "tanggal": "2024-11-13T02:33:33.446Z"
        },
        {
            "idPermohonan": "rn6ex9jqq67rqt4n9arngwo7",
            "namaResponden": "Respondent User 5",
            "namaPerusahaan": "Company DA",
            "status": "menunggu konfirmasi",
            "tanggal": "2024-11-13T02:33:33.446Z"
        },
        {
            "idPermohonan": "t6n3zxjhmd5kho6fxf56kv0a",
            "namaResponden": "Niko",
            "namaPerusahaan": "PT Test",
            "status": "dalam pengerjaan",
            "tanggal": "2024-11-16T02:04:39.144Z"
        },
        {
            "idPermohonan": "rv7x7jw96p308hmnjqix6k2x",
            "namaResponden": "Respondent User 2",
            "namaPerusahaan": "Company C",
            "status": "diterima",
            "tanggal": "2024-11-13T02:33:33.445Z"
```

```
        },
        {
            "idPermohonan": "vef0z4ymzgo8aion0tfrg0jg",
            "namaResponden": "Respondent User 3",
            "namaPerusahaan": "Company DD",
            "status": "diterima",
            "tanggal": "2024-11-13T02:33:33.445Z"
        },
        {
            "idPermohonan": "gwatog0wj1ibea60bto9q2mw",
            "namaResponden": "Respondent User 4",
            "namaPerusahaan": "Company FS",
            "status": "diterima",
            "tanggal": "2024-11-13T02:33:33.446Z"
        },
        {
            "idPermohonan": "erlvkvf7d5d7bh65b77chtcr",
            "namaResponden": "Respondent User 5",
            "namaPerusahaan": "Company DA",
            "status": "diterima",
            "tanggal": "2024-11-13T02:33:33.446Z"
        },
        {
            "idPermohonan": "sis6qqvhlo98p1m8aisazoi8",
            "namaResponden": "Respondent User 3",
            "namaPerusahaan": "Company DD",
            "status": "ditolak",
            "tanggal": "2024-11-13T02:33:33.445Z"
        }
    ],
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Assessment Request Management List Data
401	Unauthorized	Unauthorized

Tabel 96. *Status Code Assessment Request Management List*

gg. Assessment Request Management by ID

Endpoint ini digunakan untuk menampilkan pertanyaan berdasarkan ID yang dicari.

Route	/assessmentRequestManagement/{id}
Method	GET
Request	Bearer Token

Tabel 97. *Route Assessment Request Management by ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 98. *Parameter Assessment Request Management by ID*

- Response


```
{
    "tanggal": "2024-11-16T02:04:39.144Z",
    "nama": "Niko",
    "posisi": "Developer",
    "pengalamanKerja": "3 Tahun",
    "email": "niko@gmail.com",
    "namaPerusahaan": "PT Test",
    "alamat": "Jombang",
    "nomorTelepon": "43403535939",
    "username": "niko123",
    "status": "dalam pengerjaan"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Assessment Request Management Data by ID
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 99. *Status Code Assessment Request Management by ID*

hh. Update Assessment Request Management

Endpoint ini digunakan untuk memperbarui atau mengedit data manajemen permohonan asesmen ke dalam aplikasi.

Route	/assessmentRequestManagement/{id}
Method	PATCH
Request	- Bearer Token - JSON raw

Tabel 100. *Route Update Assessment Request Management*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses

			endpoint yang membutuhkan otorisasi.
assessmentId	body	String	ID milik asesmen yang berkaitan dengan manajemen permohonan asesmen yang ingin dimasukkan
createdAt	body	Timestamp	Tanggal akan secara otomatis terisi sesuai dengan tanggal dan waktu mengisi data.
name	body	String	Berisikan nama user yang bersangkutan.
username	body	String	Berisi rangkaian huruf dan atau tanpa angka yang dibuat user sebagai identitas akun
email	body	String	Alamat email yang valid dan digunakan untuk keperluan komunikasi atau otentikasi user.
password	body	String	Berisi kata sandi yang digunakan untuk mengamankan akses ke akun user.
status	body	Enum	Berisi beberapa status yang digunakan untuk validasi jika ingin melakukan asesmen ("menunggu konfirmasi", "diterima", "ditolak", "selesai", "belum diverifikasi", "dalam pengerjaan")

Tabel 101. *Parameter Update Assessment Request Management*

- Response

```
{
    "message": "Status assessment berhasil diperbarui."
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Status assessment berhasil diperbarui
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 102. *Status Code Update Assessment Request Management*

ii. Aspect and Sub Aspect on Assessment

Endpoint ini digunakan untuk menampilkan daftar aspek dan sub aspek dalam aplikasi.

Route	/assessments/aspect
Method	GET
Request	Bearer Token

Tabel 103. *Route Aspects and Sub Aspects on Assessment List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang

			dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 104. *Parameter Aspects and Sub Aspects on Assessment List*

- Response


```
{
    "data": [
        {
            "id": "gsgk0aic5n4ef8lbyq8covli",
            "name": "Deteksi",
            "createdAt": "2024-12-11T07:10:23.034Z",
            "updatedAt": "2024-12-11T07:10:23.034Z",
            "subAspects": [
                {
                    "id": "zli3ba1xl5ajdl0dt6rgljgr",
                     "name": "Mengelola deteksi Peristiwa
Siber",
                    "questionCount": "7"
                },
                {
                    "id": "hn58dzqkdb42clnlv0khlm42",
                        "name": "Memantau Peristiwa Siber
berkelanjutan",
                    "questionCount": "7"
                },
                {
                    "id": "jz9cfwbudje5q1zddqctmyyz",
                       "name": "Menganalisis anomali dan
Peristiwa Siber",
                    "questionCount": "5"
                }
            ]
        },
        {
            "id": "t6vn45nmrvv29ezpplg1jk6d",
            "name": "Identifikasi",
            "createdAt": "2024-12-11T07:10:23.034Z",
            "updatedAt": "2024-12-11T07:10:23.034Z",
            "subAspects": [
                {
```

```
                    "id": "ib08n5ahgrg72mzrtyp725vb",
                    "name": "Mengelola aset informasi",
                    "questionCount": "8"
                },
                {
                    "id": "xw3bazdspo5c249pjisgli4f",
                         "name": "Mengelola risiko rantai
pasok",
                    "questionCount": "12"
                },
                {
                    "id": "xotxn9r85fz3wka0st7xnodg",
                      "name": "Mengidentifikasi Peran dan
tanggung jawab organisasi",
                    "questionCount": "5"
                },
                {
                    "id": "y6n8hb2jkj9c982c7uymelok",
                              "name": "Menyusun strategi,
kebijakan, dan prosedur Pelindungan IIV",
                    "questionCount": "7"
                },
                {
                    "id": "gz10yc15ec95ioc83d88aft4",
                     "name": "Menilai dan mengelola risiko
Keamanan Siber",
                    "questionCount": "19"
                }
            ]
        },
        {
            "id": "p3n3sfyhncza9gcgkwc7gmaq",
            "name": "Penanggulangan dan Pemulihan",
            "createdAt": "2024-12-11T07:10:23.034Z",
            "updatedAt": "2024-12-11T07:10:23.034Z",
            "subAspects": [
                {
                    "id": "q3o1wwp92by25xg3pb387nio",
                           "name": "Meningkatkan keamanan
setelah terjadinya Insiden Siber",
                    "questionCount": "7"
                },
                {
                    "id": "w687hi0ezuswyyuivqka2rjc",
```

```
                        "name": "Menyusun perencanaan
penanggulangan dan pemulihan Insiden Siber",
                    "questionCount": "14"
                },
                {
                    "id": "mbygz73jok9hgtsjq2ofounk",
                     "name": "Menganalisis dan melaporkan
Insiden Siber",
                    "questionCount": "6"
                },
                {
                    "id": "agxhz2oho3n2lv4t4dlfi3s8",
                     "name": "Melaksanakan penanggulangan
dan pemulihan Insiden Siber",
                    "questionCount": "22"
                }
            ]
        },
        {
            "id": "yww407bnba9m7ec9njyjbzxo",
            "name": "Proteksi",
            "createdAt": "2024-12-11T07:10:23.034Z",
            "updatedAt": "2024-12-11T07:10:23.034Z",
            "subAspects": [
                {
                    "id": "lsdk38d6e2v1hz6x1t4yjtio",
                    "name": "Melindungi data",
                    "questionCount": "15"
                },
                {
                    "id": "z885tsb3rb919z1gy8ryihsf",
                    "name": "Melindungi aset fisik",
                    "questionCount": "9"
                },
                {
                    "id": "uu7kltn68ap5r5ae7yqk3mop",
                            "name": "Mengelola identitas,
autentikasi, dan kendali akses",
                    "questionCount": "6"
                },
                {
                    "id": "guusrqoipg99wd6dkhrrrw9u",
                    "name": "Melindungi jaringan",
                    "questionCount": "13"
```

```
                },
                {
                    "id": "b0t8z03qxi6uct0t2zd1y59k",
                    "name": "Melindungi sumber daya
manusia",
                    "questionCount": "13"
                },
                {
                    "id": "kruez3nrucxok4lhsofmct1q",
                    "name": "Melindungi aplikasi",
                    "questionCount": "6"
                }
            ]
        }
    ],
    "_metadata": {
        "currentPage": 0,
        "totalPages": 1,
        "totalItems": 4,
        "perPage": 10
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Aspects and Sub Aspects List Data
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 105. *Status Code Aspects and Sub Aspects on Assessment List*

jj. Questions and Options that relate to Sub Aspects and Aspects on Assessment

Endpoint ini digunakan untuk menampilkan data pertanyaan dan pilihan jawaban sesuai dengan aspek dan sub aspek yang dicari..

Route	/assessments/getAllQuestions
Method	GET
Request	Bearer Token

Tabel 106. *Route Questions and Options that relate to Sub Aspects and Aspects on Assessment List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 107. *Parameter Questions and Options that relate to Sub Aspects and Aspects on Assessment List*

- Response


```
{
            "questionId": "yv2nsuuws3bo50jgubbghdug",
                    "questionText": "Pimpinan organisasi menetapkan keamanan siber sebagai prioritas di organisasi dalam bentuk kebijakan atau komitmen pimpinan yang sesuai dengan kondisi bisnis/layanan dan operasional organisasi.",
            "needFile": false,
            "aspectsId": "ebh8uv2dwh8hslykrkwuaezg",
            "aspectsName": "Identifikasi",
            "subAspectId": "blcxc84cglmc43of407dlp5v",
             "subAspectName": "Mengidentifikasi Peran dan tanggung jawab organisasi",
            "options": [
                {
```

```
                                                "optionId":
"ofxcgkfhdqpaqqycf8htyupx",
                        "optionText": "Belum ada penerapan
kebijakan atau komitmen pimpinan terkait keamanan
siber.",
                        "optionScore": 0
                    },
                    {
                                                "optionId":
"a22334r3jm6xd4oybz0o3559",
                         "optionText": "Kebijakan Keamanan
Siber diterapkan pada sebagian kecil aspek, namun belum
ditetapkan secara formal",
                        "optionScore": 1
                    },
                    {
                                                "optionId":
"tj8izxftsvx9jd54fh5pmkvs",
                         "optionText": "Kebijakan Keamanan
Siber diterapkan pada sebagian besar aspek, namun belum
ditetapkan secara formal",
                        "optionScore": 2
                    },
                    {
                                                "optionId":
"vu7qvwzakyh4wxa1j9pc77i1",
                        "optionText": "- Kebijakan terkait
keamanan siber telah disusun dan ditetapkan - Pimpinan
berkomitmen untuk memasukkan keamanan siber sebagai
prioritas organisasi.",
                        "optionScore": 3
                    },
                    {
                                                "optionId":
"u690uupfcv8wh58eegeuwr4o",
                        "optionText": "- Kebijakan terkait
keamanan siber telah disusun, ditetapkan, dimonitor, dan
dilakukan review secara berkala - Pimpinan berkomitmen
untuk memasukkan keamanan siber sebagai prioritas
organisasi.",
                        "optionScore": 4
                    },
                    {
```

```
                                                    "optionId":
"xkn41j65ohjloykqru6u8x5k",
                        "optionText": "- Kebijakan terkait
keamanan siber telah disusun, ditetapkan, dimonitor, dan
dilakukan review serta pembaruan secara berkala -
Pimpinan berkomitmen untuk memasukkan keamanan siber
sebagai prioritas organisasi.",
                    "optionScore": 5
                }
            ]
        },
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Questions and Options List Data
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 108. *Status Code Aspects and Sub Aspects on Assessment List*

kk. Answers Data by Assessment ID on Assessment

Endpoint ini digunakan untuk menampilkan data jawaban berdasarkan Assessment ID yang dicari..

Route	/assessments/getAnswers?assessmentId={id}
Method	GET
Request	Bearer Token

Tabel 109. *Route Answers by Assessment ID*

- Parameters

Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 110. *Parameter Answers by Assessment ID*

- Response


```
{
            "id": "k4kvlp4zye3t7t3e0hhl0juo",
            "assessmentId": "azqt7ch4s8mbjx2r900s6gkh",
            "questionId": "vme8fqaz1zukr3fv8otxvp90",
            "optionId": "qoedfj3yh7hob851w1d19fnj",
            "isFlagged": false,
            "filename": null,
            "validationInformation": ""
        },
        {
            "id": "dh0qplgph0hha3wa94tv0pr3",
            "assessmentId": "azqt7ch4s8mbjx2r900s6gkh",
            "questionId": "nedigot0162k6yj34sz6iw8a",
            "optionId": "c6i5bokvk9e19ae2u2ioat2v",
            "isFlagged": false,
            "filename": null,
            "validationInformation": ""
        },
        {
            "id": "tva57x9jwgt1z3g6cs94z3y5",
            "assessmentId": "azqt7ch4s8mbjx2r900s6gkh",
            "questionId": "vy7t69s5r3xsv5pee5shg45d",
            "optionId": "jl83bojtpgaaufcl01y0rzs4",
            "isFlagged": false,
            "filename": null,
            "validationInformation": ""
        },
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Answers by assessment ID
401	Unauthorized	Unauthorized
404	Not Found	Not Found

Tabel 111. *Status Code Answers by Assessment ID*

ll. Update Toggle Flags

Endpoint ini digunakan untuk memperbarui atau mengedit data toggle flag ke dalam aplikasi.

Route	/assessments/toggleFlag
Method	PATCH
Request	- Bearer Token - JSON raw

Tabel 112. *Route Update Toggle Flags*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

assessmentId	body	String	ID milik asesmen yang berkaitan dengan pertanyaan asesmen yang ingin dimasukkan.
questionId	body	String	ID milik pertanyaan yang berkaitan dengan ID asesmen yang ingin dimasukkan.
isFlagged	body	Boolean	Berisikan data yang hanya berisi “true” atau “false”.

Tabel 113. *Parameter Update Toggle Flags*

- Response


```
{
    "message": "Flag changed successfully",
    "answer": {
        "id": "a0wnozrqbu1uqw69s8up1q8e",
        "questionId": "xxi776mrre76x3mv068oapwy",
        "optionId": null,
        "assessmentId": "azqt7ch4s8mbjx2r900s6gkh",
        "isFlagged": true,
        "filename": null,
        "validationInformation": "",
        "createdAt": "2024-12-11T12:01:46.657Z",
        "updatedAt": "2024-12-11T12:01:46.657Z"
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Flag changed successfully
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your

		input and try again.

Tabel 114. *Status Code Update Toggle Flags*

mm. Create Upload File

Endpoint ini digunakan untuk menambahkan file oleh pengguna.

Route	/assessments/uploadFile?assessmentId={id}&questionId={id}
Method	POST
Request	- Bearer Token - JSON raw

Tabel 115. *Route Create Upload File*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
assessmentId	body	String	ID asesmen yang dilakukan untuk mengerjakan asesmen.
questionId	body	String	ID pertanyaan yang bersangkutan dengan ID asesmen yang ingin dikerjakan.
answerId	form-data	String	ID jawaban yang bersangkutan dengan ID asesmen dan

			pertanyaan yang ingin dikerjakan.
filename	form -data	String	Berupa sebuah file yang ingin dimasukkan ke dalam sebuah asesmen.

Tabel 116. *Parameter Create Upload File*

- Response

```
{
    "success": true,
    "imageUrl": "/files/1735270457347-1719802439946.pdf"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Created	success: true
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 117. *Status Code Create Upload File*

nn. Create Submit Option

Endpoint ini digunakan untuk memasukkan jawaban.

Route	/assessments/submitOption
Method	POST
Request	- Bearer Token - JSON raw

Tabel 118. *Route Create Submit Option*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
assessmentId	body	String	ID asesmen yang dilakukan untuk mengerjakan asesmen.
questionId	body	String	ID pertanyaan yang bersangkutan dengan ID asesmen yang ingin dikerjakan.
optionId	body	String	ID jawaban yang bersangkutan dengan ID asesmen dan pertanyaan yang ingin dikerjakan.

Tabel 119. *Parameter Create Submit Option*

- Response


```
{
    "message": "Option submitted successfully",
    "answer": {
        "id": "dgghibj1gw0zajumuhxh5okp",
        "questionId": "hahfw9yg1wdkilvndzvtpkf2",
        "optionId": "d3l9tf169daoimb3t3g63yk1",
        "assessmentId": "azqt7ch4s8mbjx2r900s6gkh",
        "isFlagged": false,
        "filename": "1735270457347-1719802439946.pdf",
        "validationInformation": "",
        "createdAt": "2024-12-11T12:01:46.657Z",
```

```
        "updatedAt": "2024-12-27T03:34:17.354Z"
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Created	Option submitted successfully
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 120. *Status Code Create Submit Option*

oo. Create and Update Submit Validation

Endpoint ini digunakan untuk memasukkan keterangan dari jawaban yang dimasukkan.

Route	/assessments/submitValidation
Method	POST
Request	- Bearer Token - JSON raw

Tabel 121. *Route Create and Update Submit Validation*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang

			dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
assessmentId	body	String	ID asesmen yang dilakukan untuk mengerjakan asesmen.
questionId	body	String	ID pertanyaan yang bersangkutan dengan ID asesmen yang ingin dikerjakan.
validationInformation	body	String	Berisi keterangan validasi untuk mendukung jawaban yang dipilih.

Tabel 122. *Parameter Create and Update Submit Validation*

- Response


```
{
          "message": "Validation information updated successfully",
    "answer": {
        "id": "dgghibj1gw0zajumuhxh5okp",
        "questionId": "hahfw9yg1wdkilvndzvtpkf2",
        "optionId": "d3l9tf169daoimb3t3g63yk1",
        "assessmentId": "azqt7ch4s8mbjx2r900s6gkh",
        "isFlagged": false,
        "filename": "1735270457347-1719802439946.pdf",
          "validationInformation": "Penggunaan SOP tahun 2024",
        "createdAt": "2024-12-11T12:01:46.657Z",
        "updatedAt": "2024-12-27T04:03:32.219Z"
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message

200	Created	Validation Infromation updated successfully
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 123. *Status Code Create and Update Submit Validation*

pp. Update Submit Assessment

Endpoint ini digunakan untuk mengirim asesmen dan merubah status menjadi belum diverifikasi ke dalam aplikasi.

Route	/submitAssessments/{id}
Method	PATCH
Request	- Bearer Token - JSON raw

Tabel 124. *Route Update Submit Assessments*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
assessmentId	body	String	ID milik asesmen yang berkaitan dengan pertanyaan asesmen yang ingin dimasukkan.

Tabel 125. *Parameter Update Submit Assessments*

- Response

```
{
    "message": "Status assessment berhasil diperbarui."
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Status assessment berhasil diperbarui
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 126. *Status Code Update Submit Assessments*

qq. Sub Aspects average score By Assessment ID

Endpoint ini digunakan untuk menampilkan hasil rata-rata skor sub aspek berdasarkan ID asesmen yang dicari.

Route	/assessments/average-score/sub-aspects/assessments{id}
Method	GET
Request	Bearer Token

Tabel 127. *Route Sub Aspect Average Score by Assessment ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 128. *Parameter Sub Aspect Average Score by Assessment ID*

- Response


```
{
    "assessmentId": "azqt7ch4s8mbjx2r900s6gkh",
    "subAspects": [
        {
            "subAspectId": "blcxc84cglmc43of407dlp5v",
             "subAspectName": "Mengidentifikasi Peran dan
tanggung jawab organisasi",
            "averageScore": "2.3333333333333333",
            "aspectId": "ebh8uv2dwh8hslykrkwuaezg"
        },
        {
            "subAspectId": "udnbro0m900atv4tm167u338",
                    "subAspectName": "Menyusun strategi,
kebijakan, dan prosedur Pelindungan IIV",
            "averageScore": "2.5714285714285714",
            "aspectId": "ebh8uv2dwh8hslykrkwuaezg"
        }
    ]
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Sub Aspect Average Score by Assessment ID

401	Unauthorized	Unauthorized

Tabel 129. *Status Code Sub Aspect Average Score by Assessment ID*

rr. Aspects average score By Assessment ID

Endpoint ini digunakan untuk menampilkan hasil rata-rata skor sub aspek berdasarkan ID asesmen yang dicari.

Route	/assessments/average-score/aspects/assessments{id}
Method	GET
Request	Bearer Token

Tabel 130. *Route Aspect Average Score by Assessment ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 131. *Parameter Sub Aspect Average Score by Assessment ID*

- Response


```
{
    "assessmentId": "azqt7ch4s8mbjx2r900s6gkh",
    "aspects": [
        {
            "aspectId": "ebh8uv2dwh8hslykrkwuaezg",
            "aspectName": "Identifikasi",
            "averageScore": "2.5000000000000000",
            "subAspects": [
                {
```

```
                                                "subAspectId":
"blcxc84cglmc43of407dlp5v",
                        "subAspectName": "Mengidentifikasi
Peran dan tanggung jawab organisasi",
                    "averageScore": "2.3333333333333333"
                },
                {
                                                "subAspectId":
"udnbro0m900atv4tm167u338",
                    "subAspectName": "Menyusun strategi,
kebijakan, dan prosedur Pelindungan IIV",
                    "averageScore": "2.5714285714285714"
                }
            ]
        }
    ]
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Aspect Average Score by Assessment ID
401	Unauthorized	Unauthorized

Tabel 132. *Status Code Aspect Average Score by Assessment ID*

ss. Update Option

Endpoint ini digunakan untuk merubah pilihan jawaban.

Route	/assessments/updateOption
Method	PATCH
Request	- Bearer Token - JSON raw

Tabel 133. *Route Update Option*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
assessmentId	body	String	ID asesmen yang dilakukan untuk mengerjakan asesmen.
questionId	body	String	ID pertanyaan yang bersangkutan dengan ID asesmen yang ingin dikerjakan.
optionId	body	String	ID jawaban yang bersangkutan dengan ID asesmen dan pertanyaan yang ingin dikerjakan.

Tabel 134. *Parameter Update Option*

- Response

```
{
    "message": "Revision updated successfully"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message

200	Created	Revision Updated successfully
401	Unauthorized	Unauthorized
422	Unprocessable Entity	Validation failed. Please check your input and try again.

Tabel 135. *Status Code Update Option*

tt. Assessment Results List

Endpoint ini digunakan untuk menampilkan daftar hasil asesmen yang ada di dalam aplikasi.

Route	/assessmentResult
Method	GET
Request	Bearer Token

Tabel 136. *Route Assessment Results List*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 137. *Parameter Assessment Results List*

- Response


```
{
    "data": [
        {
            "id": "ylyqvgrgc5utnml3y82jedd7",
            "respondentName": "asdfghjkl",
```

```
            "companyName": "polinema",
            "statusAssessments": "belum diverifikasi",
            "statusVerification": "belum diverifikasi",
            "assessmentsResult": "4.45"
        },
            . . .,
        {
            "id": "wj2ftpr4zejykhg8q2glyxp4",
            "respondentName": "Respondent User 4",
            "companyName": "Company FS",
            "statusAssessments": "selesai",
            "statusVerification": "sudah diverifikasi",
            "assessmentsResult": null
        }
    ],
    "_metadata": {
        "currentPage": 0,
        "totalPages": 1,
        "totalItems": "9",
        "perPage": 40
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Assessment results List Data
401	Unauthorized	Unauthorized

Tabel 138. *Status Code Assessment Results List*

uu. Assessment Result by ID

Endpoint ini digunakan untuk menampilkan hasil asesmen yang ada di dalam aplikasi berdasarkan ID milik data asesmen.

Route	/assessmentResult/{id}
Method	GET

Request	Bearer Token

Tabel 139. *Route Assessment Results by ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 140. *Parameter Assessment Results by ID*

- Response

```
{
    "respondentName": "asdfghjkl",
    "position": "supervisor",
    "workExperience": "2 Tahun",
    "email": "aaa@gmail.com",
    "companyName": "polinema",
    "address": "Jl. Kembang Turi, Lowokwaru Malang",
    "phoneNumber": "08988834242",
    "username": "aaa",
    "assessmentDate": "2024-12-11T07:22:59.234Z",
    "statusAssessment": "belum diverifikasi",
    "statusVerification": "belum diverifikasi",
    "assessmentsResult": "4.45"
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message

200	Ok	Assessment result by ID
401	Unauthorized	Unauthorized

Tabel 141. *Status Code Assessment Results by ID*

vv. Verified Assessment Result by ID

Endpoint ini digunakan untuk menampilkan hasil asesmen yang telah terverifikasi dan berdasarkan ID milik data asesmen.

Route	/assessmentResult/verified/{id}
Method	GET
Request	Bearer Token

Tabel 142. *Route Verified Assessment Results by ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat superadmin berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 143. *Parameter Verified Assessment Results by ID*

- Response


```
{
    "respondentName": "Respondent User 2",
    "position": "Employee",
    "workExperience": "4 years",
    "email": "respondentUser2@gmail.com",
    "companyName": "Company C",
    "address": "Address D",
    "phoneNumber": "1234567890123",
```

```
    "username": "respondentUser2",
    "assessmentDate": "2024-12-11T07:10:28.026Z",
    "statusAssessment": "selesai",
    "statusVerification": "sudah diverifikasi",
    "verifiedAssessmentsResult": null
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Assessment result by ID
401	Unauthorized	Unauthorized

Tabel 144. *Status Code Verified Assessment Results by ID*

ww. Get All Question by Assessment ID

Endpoint ini digunakan untuk menampilkan semua pertanyaan yang terkait dalam suatu asesmen berdasarkan asesmen ID.

Route	/assessmentResult/getAllQuestion/{id}
Method	GET
Request	Bearer Token

Tabel 145. *Route Get All Question by Assessment ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint

			yang membutuhkan otorisasi.

Tabel 146. *Parameter Get All Question by Assessment ID*

- Response


```
{
    "data": [
        {
            "questionId": "eb4wr9um6h0oieoixlg2pd1z",
                    "questionText": "Pimpinan organisasi menetapkan keamanan siber sebagai prioritas di organisasi dalam bentuk kebijakan atau komitmen pimpinan yang sesuai dengan kondisi bisnis/layanan dan operasional organisasi.",
            "needFile": false,
            "aspectsId": "t6vn45nmrvv29ezpplg1jk6d",
            "aspectsName": "Identifikasi",
            "subAspectId": "xotxn9r85fz3wka0st7xnodg",
             "subAspectName": "Mengidentifikasi Peran dan tanggung jawab organisasi",
            "options": [
                {
                                                    "optionId": "santt6a1yubwijdr11oq8olj",
                        "optionText": "Belum ada penerapan kebijakan atau komitmen pimpinan terkait keamanan siber.",
                    "optionScore": 0
                },
                {
                                                    "optionId": "qniwex74us8pg4s4qbr71ni5",
                        "optionText": "Kebijakan Keamanan Siber diterapkan pada sebagian kecil aspek, namun belum ditetapkan secara formal",
                    "optionScore": 1
                },
                {
                                                    "optionId": "grpgddrdveueyio19dr5as8k",
                        "optionText": "Kebijakan Keamanan Siber diterapkan pada sebagian besar aspek, namun belum ditetapkan secara formal",
                    "optionScore": 2
```

```
                },
                {
                                                    "optionId":
"peauqlatn3jfy8ewnq5ry3xc",
                        "optionText": "- Kebijakan terkait
keamanan siber telah disusun dan ditetapkan - Pimpinan
berkomitmen untuk memasukkan keamanan siber sebagai
prioritas organisasi.",
                    "optionScore": 3
                },
                {
                                                    "optionId":
"xzbabggn0xe5tyo6s6srwl75",
                        "optionText": "- Kebijakan terkait
keamanan siber telah disusun, ditetapkan, dimonitor, dan
dilakukan review secara berkala - Pimpinan berkomitmen
untuk memasukkan keamanan siber sebagai prioritas
organisasi.",
                    "optionScore": 4
                },
                {
                                                    "optionId":
"m1vzcnym21qna4n5z6ez20v9",
                        "optionText": "- Kebijakan terkait
keamanan siber telah disusun, ditetapkan, dimonitor, dan
dilakukan review serta pembaruan secara berkala -
Pimpinan berkomitmen untuk memasukkan keamanan siber
sebagai prioritas organisasi.",
                    "optionScore": 5
                }
            ]
        },
                . . .,
    ]
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message

200	Ok	All Questions and Options Data
404	Not Found	Not Found

Tabel 147. *Status Code Get All Question by Assessment ID*

xx. Get Sub Aspects Average Score by Assessment ID

Endpoint ini digunakan untuk menampilkan nilai rata-rata semua sub aspek dalam suatu asesmen berdasarkan asesmen ID.

Route	/assessmentResult/average-score/sub-aspects/assessments/{assessmentId}
Method	GET
Request	Bearer Token

Tabel 148. *Route Get Sub Aspects Average Score by Assessment ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 149. *Parameter Get Sub Aspects Average Score by Assessment ID*

- Response


```
{
    "assessmentId": "ylyqvgrgc5utnml3y82jedd7",
    "subAspects": [
        {
            "subAspectId": "agxhz2oho3n2lv4t4dlfi3s8",
            "subAspectName": "Melaksanakan penanggulangan
dan pemulihan Insiden Siber",
```

```
            "averageScore": "4.1818181818181818",
            "aspectId": "p3n3sfyhncza9gcgkwc7gmaq"
        },
              . . .,
        {
            "subAspectId": "zli3ba1xl5ajdl0dt6rgljgr",
            "subAspectName": "Mengelola deteksi Peristiwa
Siber",
            "averageScore": "5.0000000000000000",
            "aspectId": "gsgk0aic5n4ef8lbyq8covli"
        }
    ]
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Sub Aspects Average Score
404	Not Found	Not Found

Tabel 150. *Status Code Get Sub Aspects Average Score by Assessment ID*

yy. Get Aspects Average Score by Assessment ID

Endpoint ini digunakan untuk menampilkan nilai rata-rata semua aspek dalam suatu asesmen berdasarkan asesmen ID.

Route	/assessmentResult/average-score/aspects/assessments/{assessmentId}
Method	GET
Request	Bearer Token

Tabel 151. *Route Get Aspects Average Score by Assessment ID*

- Parameters

Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 152. *Parameter Get Aspects Average Score by Assessment ID*

- Response


```
{
    "assessmentId": "ylyqvgrgc5utnml3y82jedd7",
    "aspects": [
        {
            "aspectId": "gsgk0aic5n4ef8lbyq8covli",
            "aspectName": "Deteksi",
            "averageScore": "4.1666666666666667",
            "subAspects": [
                {
                                                "subAspectId":
"hn58dzqkdb42clnlv0khlm42",
                     "subAspectName": "Memantau Peristiwa
Siber berkelanjutan",
                    "averageScore": "3.8571428571428571"
                },
                {
                                                "subAspectId":
"jz9cfwbudje5q1zddqctmyyz",
                              "subAspectName": "Menganalisis
anomali dan Peristiwa Siber",
                    "averageScore": "3.6000000000000000"
                },
                {
                                                "subAspectId":
"zli3ba1xl5ajdl0dt6rgljgr",
                      "subAspectName": "Mengelola deteksi
Peristiwa Siber",
                    "averageScore": "5.0000000000000000"
                }
```

```
            ]
        },
            . . .,
    ]
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Aspects Average Score
404	Not Found	Not Found

Tabel 153. *Status Code Get Aspects Average Score by Assessment ID*

zz. Get Answers by Assessment ID

Endpoint ini digunakan untuk menampilkan semua jawaban dalam suatu asesmen berdasarkan asesmen ID.

Route	/assessmentResult/getAnswers/{assessmentId}
Method	GET
Request	Bearer Token

Tabel 154. *Route Get Answers by Assessment ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 155. *Parameter Get Answers by Assessment ID*

- Response


```
{
    "data": [
        {
            "id": "x3mi625n2wr3kz242ggoo9xe",
            "answerId": "zovasjeo6rfa23vlrexcmg3v",
            "newOptionId": "1px5t25heezc5q1b8q0wcoxa",
            "newValidationInformation": "",
            "questionId": "okawj997k8qx996ftayb75un"
        },
              . . .,
        }
            "id": "f1jumnnvjn3hlx9hm9n8cb8c",
            "answerId": "qdd3z9xlyz75t7duo87lu1gi",
            "newOptionId": "mq85mgoq1kgkcb8kpw0hf0x3",
            "newValidationInformation": "",
            "questionId": "hr8mh9sfrji14y2s5y3w0onj"
        }
    ]
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	All Answers Data
404	Not Found	Not Found

Tabel 156. *Status Code Get Answers by Assessment ID*

aaa. Create Answer Revisions

Endpoint ini membuat data di database yang digunakan untuk menampung revisi jawaban berdasarkan asesmen ID.

Route	/assessmentResult/answer-revisions
Method	POST

Request	- Bearer Token - JSON raw

Tabel 157. *Route Create Answer Revisions by Assessment ID*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.
assessmentId	body	String	ID yang dimiliki tiap asesmen
revisedBy	body	String	Nama user yang melakukan proses verifikasi

Tabel 158. *Parameter Create Answer Revisions by Assessment ID*

- Response


```
{
    "message": "Answer revisions created successfully",
    "data": [
        {
            "id": "wqd588velo8zxeehoh42sd6n",
            "answerId": "zovasjeo6rfa23vlrexcmg3v",
            "newOptionId": "1px5t25heezc5q1b8q0wcoxa",
            "revisedBy": "superadmin",
            "newValidationInformation": "",
            "createdAt": "2024-12-27T10:25:24.883Z"
        },
              . . .,
        {
            "id": "q6kljvzi31hx079ueynwhh47",
            "answerId": "qdd3z9xlyz75t7duo87lu1gi",
```

```
            "newOptionId": "mq85mgoq1kgkcb8kpw0hf0x3",
            "revisedBy": "superadmin",
            "newValidationInformation": "",
            "createdAt": "2024-12-27T10:25:24.883Z"
        }
    ]
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
201	Created	Answer revision created successfully
404	Not Found	Not Found

Tabel 159. *Status Code Create Answer Revisions by Assessment ID*

bbb. Add Assessment Result Verified Status

Endpoint ini digunakan untuk mengubah status hasil asesmen menjadi terverifikasi berdasarkan asesmen ID.

Route	/assessmentResult/{id}
Method	PATCH
Request	Bearer Token

Tabel 160. *Route Add Assessment Result Verified Status*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang

			dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 161. *Parameter Add Assessment Result Verified Status*

- Response

```
{
    "message": "Assessment berhasil diverifikasi",
    "data": [
        {
            "id": "ylyqvgrgc5utnml3y82jedd7",
            "respondentId": "kld83ruwf5mne39hol45p23q",
            "status": "belum diverifikasi",
            "reviewedBy": "Super Admin",
            "reviewedAt": "2024-12-11T07:24:14.017Z",
            "verifiedBy": null,
            "verifiedAt": null,
            "createdAt": "2024-12-11T07:22:59.234Z"
        }
    ]
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Assessment verified
404	Not Found	Not Found

Tabel 162. *Status Code Add Assessment Result Verified Status*

ccc. Change Assessment Result Status to Done

Endpoint ini digunakan untuk mengubah status hasil asesmen menjadi selesai berdasarkan asesmen ID.

Route	/assessmentResult/submitAssessmentRevision/{id}
Method	PATCH

Request	Bearer Token

Tabel 163. *Route Change Assessment Result Status to Done*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 164. *Parameter Change Assessment Result Status to Done*

- Response

```
{
    "message": "Status assessment berhasil diperbarui."
}
```

- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Assessment status updated successfully
404	Not Found	Not Found

Tabel 165. *Status Code Change Assessment Result Status to Done*

ddd. Update Validation

Endpoint ini digunakan untuk memperbarui validasi jawaban berdasarkan asesmen ID.

Route	/assessmentResult/updateValidation

Method	POST
Request	Bearer Token

Tabel 166. *Route Update Validation*

- Parameters

  Pada endpoint ini, terdapat beberapa parameter yang perlu dikirimkan ke server. Dibawah ini adalah parameter yang perlu diberikan.

Parameter	Type	Data Type	Description
Token	auth	String	Token autentikasi unik yang dihasilkan saat user berhasil login untuk mengakses endpoint yang membutuhkan otorisasi.

Tabel 167. *Parameter Update Validation*

- Response


```
{
    "message": "Revision updated successfully",
    "revision": {
        "id": "sci8u0v0qygq3b19nwww5gnw",
        "answerId": "axqwgx8d10itofz7cgar6v61",
        "newOptionId": "cvuctmbx6az5p9m2d8xr2ych",
        "revisedBy": "superadmin",
            "newValidationInformation": "Uji coba UPDATE
VALIDASI",
        "createdAt": "2024-12-27T10:25:24.883Z"
    }
}
```


- Errors

  Endpoint ini menggunakan beberapa error code seperti dibawah ini.

Code	Error	Message
200	Ok	Revision update successfully

404	Not Found	Answer not found for given assessmentId and questionId

Tabel 168. *Status Code Update Validation*

6. **Logging dan error handling**

H. Deployment

1. **Panduan deployment ke server produksi**
    a. Bangun Aplikasi *Frontend*:
    `npm run build`
    b. Salin Hasil Build ke Server Produksi:
        - Gunakan SSH untuk mengakses *server*.
        - Salin folder `dist` atau `build` ke direktori *server*.
    c. Pasang *Dependency* di *Server Backend*:
    `cd backend`
    `npm install`
    d. Jalankan Aplikasi *Backend:*
    `npm run start`
2. **Pengaturan Nginx (untuk reverse proxy)**
    a. Instal Nginx di Server (jika belum ada):
    `sudo apt update`
    `sudo apt install nginx`
    b. Konfigurasi Nginx: Buka file konfigurasi Nginx:
    `sudo nano /etc/nginx/sites-available/default`
    Tambahkan konfigurasi berikut:

```
server {
    listen 80;
    server_name your_domain_or_ip;

    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}
```

c. *Restart* Nginx:

```
sudo systemctl restart nginx
```

3. Konfigurasi database pada server

a. Install Database (PostgreSQL):

```
sudo apt install postgresql
```

b. Akses PostgreSQL:

```
sudo -u postgres psql
```

c. Buat *Database* dan *User:*

```
CREATE DATABASE amati_db;
CREATE USER amati_user WITH ENCRYPTED PASSWORD 'password';
GRANT ALL PRIVILEGES ON DATABASE amati_db TO amati_user;
```

d. Konfigurasi *Environment Variable*: Tambahkan konfigurasi di file `.env`:

```
DATABASE_URL=postgresql://amati_user:password@localhost:5432/amati_db
```

e. Jalankan Migrasi *Database:*

```
cd backend
npm run db:push
```

```
npm run db:seed
```