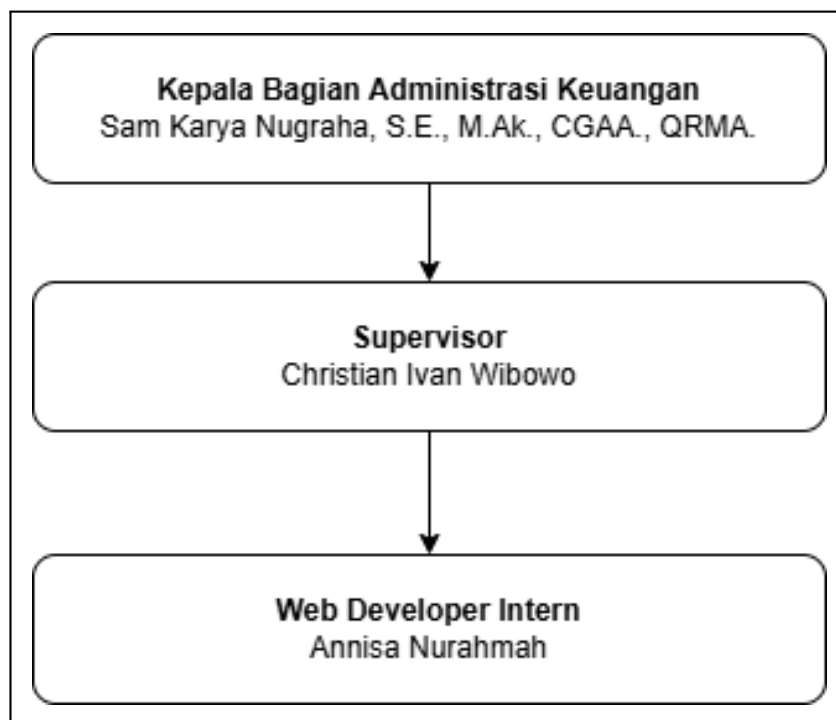


## BAB 3

### PELAKSANAAN KERJA MAGANG

#### 3.1 Kedudukan dan Koordinasi

Kegiatan magang dilaksanakan di lingkungan Sekretariat Jenderal DPR RI dengan posisi sebagai *Web Developer Intern*. Kedudukan magang berada di bawah Bagian Administrasi Keuangan, Biro Keuangan, Deputy Bidang Administrasi. Dalam pelaksanaannya, berkoordinasi dilakukan secara langsung dengan *Supervisor* dan Kepala Bagian Administrasi Keuangan yang bertindak sebagai pembimbing serta penerima hasil pekerjaan. Struktur kedudukan dan alur koordinasi magang dapat dilihat pada Gambar 3.1.



Gambar 3.1. Struktur kedudukan dan alur koordinasi Web Developer Intern di Sekretariat Jenderal DPR RI

Dalam struktur tersebut, Kepala Bagian Administrasi Keuangan berperan sebagai penanggung jawab utama yang memberikan arahan umum, menetapkan prioritas kerja, serta melakukan evaluasi terhadap hasil pengembangan sistem. *Supervisor* bertugas mengawasi pelaksanaan teknis harian, memberikan umpan balik terhadap kemajuan pekerjaan, dan memastikan hasil pengembangan sesuai kebutuhan pengguna di unit kerja. Sementara sebagai *Web Developer Intern* bertanggung jawab untuk melakukan implementasi teknis, seperti pengembangan fitur, pengujian, dan pemeliharaan modul yang berkaitan dengan sistem internal DIGITall khususnya pada modul Berita, Lembur, dan PIPK. Proses koordinasi dilakukan melalui komunikasi harian, baik melalui pesan WhatsApp maupun pertemuan tatap muka, serta rapat mingguan untuk membahas

perkembangan dan penyesuaian prioritas pekerjaan.

### 3.2 Tugas yang Dilakukan

Kegiatan magang dilaksanakan pada tim programmer pengembang sistem DIGITall Keuangan di bagian Administrasi Keuangan Sekretariat Jenderal DPR RI. Sistem DIGITall Keuangan terdiri atas berbagai modul yang saling terintegrasi untuk mendukung proses administrasi dan pelaporan keuangan. Selama periode magang, tanggung jawab pengembangan difokuskan pada tiga modul yang menjadi bagian pekerjaan, yaitu Modul Berita, Modul Lembur, dan Modul PIPK (Pengendalian Intern atas Pelaporan Keuangan). Pengembangan dilakukan menggunakan *framework* Laravel dengan basis data MySQL serta menerapkan pendekatan *role-based access control* untuk mendukung keamanan, keterhubungan data, dan efisiensi sistem.

1. Pengembangan Modul Berita dilakukan untuk memfasilitasi penyampaian informasi internal antarunit kerja di lingkungan Sekretariat Jenderal. Tahapan pelaksanaan mencakup perancangan struktur basis data (beritas, berita.kewenangan, dan berita.dismiss), integrasi dengan tabel users dan roles, serta pembuatan fitur pengaturan hak akses berdasarkan peran pengguna. Logika notifikasi otomatis diterapkan agar berita muncul pada *dashboard* tiga hari sebelum tenggat (H-3), dilengkapi mekanisme *dismiss* untuk menutup notifikasi secara mandiri. Tabel utama diperbarui dengan kolom *due\_date*, *sisahari*, dan *ditutup\_oleh* untuk mendukung pelacakan status berita. Pengujian fungsi dilakukan secara bertahap untuk memastikan seluruh alur tambah, ubah, dan hapus data berjalan sesuai rancangan sistem.
2. Pengembangan Modul Lembur dilakukan untuk mengelola proses koreksi data lembur yang dilakukan oleh Kepala Bagian (Kabag) dan pegawai terkait. Perancangan diawali dengan pembentukan relasi antar tabel pegawai, satuan\_kerja, dan daftar\_lembur. Halaman utama menampilkan daftar periode lembur beserta status pengiriman per satuan kerja, dilengkapi tombol navigasi koreksi, *preview*, dan *e-sign*. Fitur koreksi jam lembur diimplementasikan menggunakan *DataTables server-side* dengan dukungan fungsi pencarian dan filter status “Sudah/Belum Koreksi”. Validasi input diterapkan untuk mencegah kelebihan jam koreksi dan duplikasi data antarperiode. Selain itu, indikator progres, notifikasi penyimpanan, serta halaman Daftar Nominal (Dafnom) dikembangkan untuk meningkatkan kejelasan proses koreksi dan mendukung tahap implementasi tanda tangan digital. Seluruh tahapan penyusunan modul ini melibatkan proses uji coba internal untuk memastikan keterpaduan logika antarhalaman.
3. Pengembangan Modul PIPK dilakukan sebagai sistem pendukung pengendalian internal atas pelaporan keuangan dengan melibatkan tiga peran utama, yaitu Admin, Penerap, dan Approver. Peran Admin mencakup pengelolaan data dasar sistem, pengaturan akun signifikan, penetapan struktur penugasan, serta penyiapan parameter yang digunakan dalam proses penerapan dan persetujuan. Selanjutnya, Penerap berperan dalam melakukan pengisian dan penerapan data pengendalian internal pada Tabel A berdasarkan akun signifikan dan assignment yang telah ditetapkan oleh sistem. Tahapan berikutnya dilakukan oleh Approver yang melakukan verifikasi dan persetujuan terhadap data yang diajukan oleh Penerap.

### 3.3 Uraian Pelaksanaan Magang

Selama kegiatan magang berlangsung, berbagai tugas dikelompokkan berdasarkan periode kerja mingguan. Pekerjaan mencakup pengenalan lingkungan kerja, pembentukan struktur proyek, serta pengembangan tiga modul yang menjadi tanggung jawab, yaitu Modul Berita, Modul Lembur, dan Modul PIPK. Rincian aktivitas mingguan ditunjukkan pada Tabel 3.1.

Tabel 3.1. Pekerjaan yang dilakukan setiap minggu

Minggu Ke-	Pekerjaan yang dilakukan
1	Pengenalan lingkungan kerja di bagian tim <i>developer</i> Administrasi Keuangan Sekretariat Jenderal DPR RI; penjelasan aturan dan tata tertib kerja; pembuatan <i>ID card</i> ; pengenalan arsitektur sistem DIGITall, alur pengembangan, dan <i>pipeline</i> rilis modul; pengenalan gedung Sekretariat Jenderal Nusantara I–II; penyiapan <i>environment</i> pengembangan (GitHub, XAMPP, Laravel, MySQL, Composer, Docker) dan uji koneksi basis data.
2	Rapat koordinasi awal proyek DIGITall Keuangan untuk pemetaan modul prioritas (Berita, Lembur, dan PIPK) serta pembentukan tim per modul; pengenalan gedung Nusantara III dan IV; pembahasan skema repositori dan aturan <i>branching</i> Git; identifikasi kebutuhan fitur utama; penyusunan rencana kerja; serta pembuatan <i>dashboard</i> awal untuk pemanggilan data dari basis data.
3	Pengembangan awal Modul Berita meliputi perancangan arsitektur basis data (tabel beritas, berita_kewenangan, dan berita_dismiss); integrasi dengan tabel pengguna dan peran; penentuan atribut utama; pembuatan fungsi notifikasi otomatis (H-3); serta integrasi tampilan berita pada <i>slider dashboard</i> .
4	Penyempurnaan Modul Berita dengan memastikan fungsi tambah, ubah, dan hapus data berjalan dengan baik; pembuatan nomor berita otomatis; penambahan kolom <i>deadline</i> dan sisa hari; pencatatan log notifikasi; serta penyesuaian desain antarmuka dan fungsi pencarian data.
5	Finalisasi Modul Berita melalui penambahan <i>pagination</i> , perbaikan tampilan <i>pop-up</i> , dan pengujian fungsionalitas bersama pengguna; verifikasi seluruh fungsi dinyatakan sesuai dengan kebutuhan operasional.
6	Pengembangan awal Modul Lembur (peran Kepala Bagian) dengan perancangan basis data lembur; pembuatan halaman utama daftar periode lembur; integrasi relasi ke data pegawai dan satuan kerja; serta penerapan <i>DataTables server-side</i> untuk pencarian dan filter data.
7	Pengembangan halaman koreksi lembur yang menampilkan daftar pegawai per satuan kerja; formulir koreksi jam lembur; validasi input jam lembur; pembaruan status otomatis; serta pengujian sinkronisasi antar halaman.
8	Penyempurnaan Modul Lembur melalui pembuatan halaman edit per pegawai, pratinjau daftar nominal lembur, pengaktifan awal fitur <i>e-sign</i> , serta pengujian alur koreksi dari input hingga validasi akhir.
Lanjut ke halaman berikutnya	

Tabel 3.1. Pekerjaan yang dilakukan setiap minggu (lanjutan)

Minggu Ke-	Pekerjaan yang dilakukan
9	Pengembangan awal Modul PIPK (Penilai - Tabel A) dengan penyusunan rancangan basis data; penetapan relasi akun, biro, dan bagian; definisi status A dan A1; pembuatan halaman indeks penilai; serta implementasi fungsi tambah <i>assignment</i> .
10	Pengembangan lanjutan Modul PIPK (Penilai - Tabel A) dengan menampilkan daftar akun signifikan; penambahan halaman detail akun; penerapan logika pengajuan ke Approver; serta penyempurnaan tampilan tabel dan kolom aksi.
11	Implementasi Modul PIPK (Approver - Tabel A1) dengan penambahan halaman verifikasi; integrasi tampilan dokumen PDF; serta pengembangan aksi persetujuan dan penolakan dengan pembaruan status otomatis.
12	Evaluasi dan revisi akhir tahap pertama Modul PIPK melalui rapat evaluasi internal; penyempurnaan logika status PIPK Tabel A; perbaikan validasi formulir; serta persiapan struktur awal untuk pengembangan tabel lanjutan.
13	Revisi lanjutan Modul PIPK berdasarkan hasil rapat, meliputi perbaikan tata letak tabel dan penyempurnaan kolom status agar terbaru otomatis; pengujian ulang alur pengajuan dan penolakan setelah pembaruan sistem.
14	Pengembangan Modul PIPK Admin untuk penetapan kode akun dengan penambahan mekanisme persetujuan berjenjang oleh Ketua Tim Penilai dan Inspektur I; pembuatan halaman penetapan tim penilai sebagai prasyarat penambahan akun; serta pengembangan halaman approver khusus untuk persetujuan kode akun.
15	Penyempurnaan alur Modul PIPK dengan penambahan fitur berita acara penetapan kode akun; pemisahan halaman tambah akun <i>assignment</i> ; serta revisi alur Tabel A agar memiliki tahap pengajuan, verifikasi, dan <i>e-sign</i> secara berurutan.
16	Revisi lanjutan Modul PIPK dengan penambahan fitur verifikasi dan <i>e-sign</i> pada halaman Penilai dan Approver; perbaikan logika persetujuan Tabel A; pengujian ulang alur kerja; serta penyesuaian tampilan halaman detail agar informasi status dan verifikasi lebih jelas.
17	Pengujian akhir Modul PIPK Tabel A setelah seluruh revisi diterapkan; sinkronisasi alur Penilai dan Approver; serta persiapan pengembangan Tabel C untuk kebutuhan unggah dokumen pendukung oleh Penerap yang dilakukan di Tabel A.

### 3.3.1 Penyiapan Lingkungan Pengembangan

Penyiapan lingkungan pengembangan merupakan tahap awal yang penting untuk memastikan seluruh proses implementasi sistem DIGITall berjalan stabil dan konsisten. Lingkungan ini dibangun dengan menyesuaikan konfigurasi dan versi perangkat lunak yang telah digunakan pada proyek DIGITall sebelumnya. Kombinasi XAMPP, Docker, PHP 8.2, Laravel 8, serta berbagai alat



bantu diterapkan untuk menjaga kompatibilitas dan efisiensi kerja tim. Rincian perangkat *software* dan aplikasi pendukung dijelaskan sebagai berikut.

## 1. Software Pendukung

### (a) PHP 8.2

PHP 8.2 adalah *runtime environment* utama yang digunakan dalam proyek ini. Pemilihan versi ini dilakukan untuk menyesuaikan dengan konfigurasi proyek DIGITall yang telah ada sebelumnya dan dikembangkan menggunakan PHP 8.2.

### (b) Laravel 8

Laravel 8 adalah *framework* utama yang digunakan untuk membangun sistem DIGITall. Framework ini mengadopsi arsitektur *Model–View–Controller* (MVC) yang memisahkan logika bisnis, tampilan, dan data agar lebih terstruktur. Laravel juga menyediakan *Blade template engine* dan *Eloquent ORM* untuk mempermudah pembuatan antarmuka serta pengelolaan data. Penggunaan Laravel memungkinkan integrasi berbagai modul seperti Berita, Lembur, dan PIPK lebih mudah.

### (c) XAMPP (Apache Web Server)

XAMPP adalah paket pengembangan web yang menyediakan layanan Apache, PHP, dan MySQL untuk lingkungan lokal. Dalam proyek ini, XAMPP digunakan untuk menjalankan *web server* Apache yang menampilkan aplikasi Laravel. Layanan MySQL bawaan dinonaktifkan karena database dikelola melalui Docker, sehingga menghindari konflik port.

### (d) Docker Desktop

Docker Desktop adalah platform virtualisasi berbasis *container* yang digunakan untuk menjalankan MySQL dan layanan lain secara terpisah satu sama lain, sehingga setiap layanan dapat berjalan tanpa saling mengganggu.[7]. Pendekatan ini menjaga konsistensi versi dan konfigurasi di semua perangkat pengembang.

### (e) MySQL 8 (Docker Container)

MySQL 8 adalah sistem manajemen basis data relasional (RDBMS) yang digunakan untuk menyimpan seluruh data aplikasi [8], termasuk data Berita, Lembur dan PIPK. Versi ini digunakan untuk menjaga kompatibilitas dengan struktur basis data yang sudah diterapkan pada proyek DIGITall sejak tahap pengembangan awal.

## 2. Aplikasi Pendukung

### (a) Visual Studio Code

Visual Studio Code adalah *Integrated Development Environment (IDE)* yang digunakan untuk menulis, menjalankan, dan menguji kode sumber. IDE ini menyediakan berbagai ekstensi khusus Laravel seperti *Blade Snippets*, *PHP IntelliSense*, dan *Artisan*, yang mempercepat proses pengembangan. Dukungan *Language Server Protocol (LSP)* di dalam VS Code meningkatkan produktivitas dengan menyediakan fitur pelengkapan otomatis dan penelusuran kode lintas file.

### (b) Git dan GitHub

Git adalah sistem kontrol versi terdistribusi yang digunakan bersama platform

GitHub untuk mengelola perubahan kode selama proses pengembangan. GitHub mendukung kolaborasi antar anggota tim dan memudahkan pelacakan revisi kode secara historis[9].

(c) phpMyAdmin

phpMyAdmin adalah alat bantu utama dalam pengelolaan basis data. phpMyAdmin digunakan untuk manipulasi data berbasis web.

### 3.3.2 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem dilakukan untuk mengidentifikasi fungsi, batasan, dan kebutuhan teknis yang diperlukan oleh pengguna sistem. Berdasarkan hasil diskusi dengan tim dan pihak pengguna, kebutuhan sistem dibagi menjadi dua jenis yaitu kebutuhan fungsional dan nonfungsional.

#### 1. Kebutuhan Fungsional

- (a) Sistem harus menyediakan Modul Berita untuk pengelolaan informasi internal, termasuk pembuatan, pengeditan, penjadwalan, dan penghapusan berita.
- (b) Sistem harus menyediakan Modul Lembur untuk mendukung proses pengajuan, persetujuan, serta perhitungan lembur pegawai secara otomatis.
- (c) Sistem harus menyediakan Modul PIPK untuk melakukan verifikasi dan pemantauan akun signifikan yang berhubungan dengan pelaporan keuangan.
- (d) Sistem harus mendukung autentikasi dan manajemen peran pengguna seperti Admin, Kabag, dan Approver.

#### 2. Kebutuhan Nonfungsional

- (a) Sistem harus berjalan dengan stabil pada lingkungan Apache-Laravel dan basis data MySQL.
- (b) Aplikasi harus responsif dan dapat diakses pada berbagai perangkat.
- (c) Keamanan data pengguna harus dijaga melalui validasi, autentikasi, dan pembatasan akses.

### 3.3.3 Modul Admin Berita

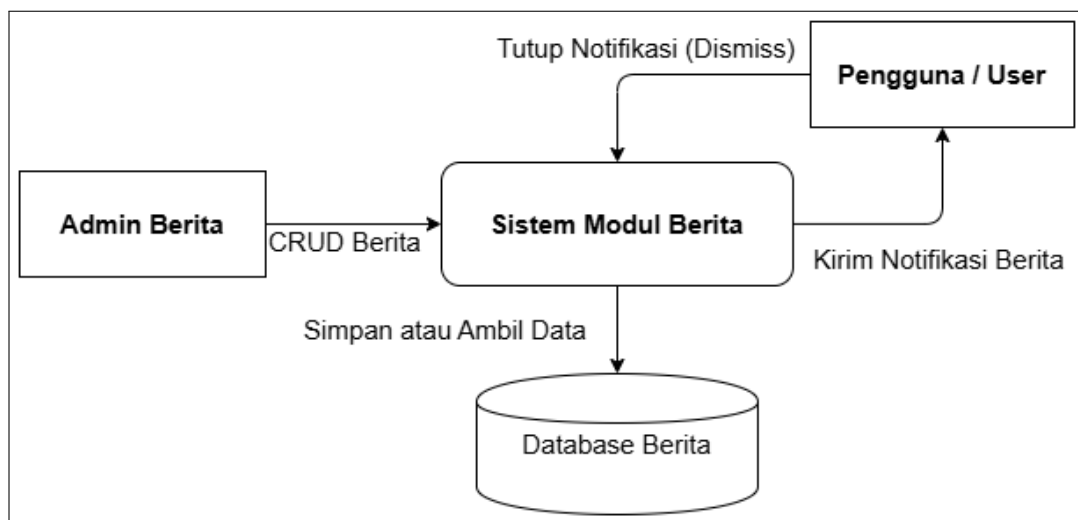
Modul Admin Berita pada sistem DIGITall digunakan untuk mengelola penyebaran informasi internal antarunit kerja di lingkungan Sekretariat Jenderal DPR RI. Melalui modul ini, admin dapat menambahkan, memperbarui, dan menghapus berita yang ditujukan kepada kelompok kewenangan tertentu, seperti bagian, biro, atau satuan kerja. Selain itu, sistem juga mencatat setiap interaksi pengguna saat notifikasi berita ditutup. Data tersebut kemudian digunakan untuk menghitung indikator kinerja dan menganalisis tingkat keterlibatan pengguna terhadap informasi yang disampaikan.

## A Data Flow Diagram (DFD)

Data Flow Diagram (DFD) digunakan untuk menggambarkan aliran data pada sistem, mulai dari masukan yang diberikan pengguna, proses yang dilakukan oleh sistem, hingga keluaran yang dihasilkan. DFD membantu memahami hubungan antarproses, penyimpanan data, serta entitas eksternal yang berinteraksi dengan sistem. Pada Modul Admin Berita, DFD dibagi menjadi dua level utama, yaitu DFD Level 0 yang menunjukkan konteks umum sistem dan DFD Level 1 yang menggambarkan rincian proses internal di dalamnya.

### 1. DFD Level 0

Gambar 3.2 menunjukkan *Data Flow Diagram* (DFD) Level 0 pada Modul Admin Berita yang menggambarkan hubungan antara sistem dengan entitas eksternal yang terlibat. Pada diagram ini, terdapat tiga entitas utama, yaitu Admin Berita, Pengguna, dan Database Berita. Admin Berita berperan dalam melakukan pengelolaan data berita, meliputi proses penambahan, pengubahan, dan penghapusan informasi berita pada sistem. Pengguna berperan sebagai penerima notifikasi berita yang dikirimkan oleh sistem serta memiliki hak untuk menutup (*dismiss*) notifikasi yang diterima. Sementara itu, Database Berita berfungsi sebagai media penyimpanan seluruh data berita, data kewenangan, serta log interaksi pengguna terkait penutupan notifikasi. Alur kerja sistem dimulai ketika Admin Berita melakukan input data berita ke dalam sistem Modul Berita. Data tersebut kemudian diproses dan disimpan ke dalam Database Berita. Selanjutnya, sistem akan menampilkan daftar berita serta mengirimkan notifikasi kepada pengguna sesuai dengan jadwal atau ketentuan yang telah ditetapkan. Ketika pengguna menutup notifikasi berita, sistem akan mencatat tindakan tersebut sebagai log penutupan yang disimpan ke dalam basis data. Dengan demikian, DFD Level 0 ini memberikan gambaran umum mengenai alur data dan interaksi antar entitas pada Modul Admin Berita.

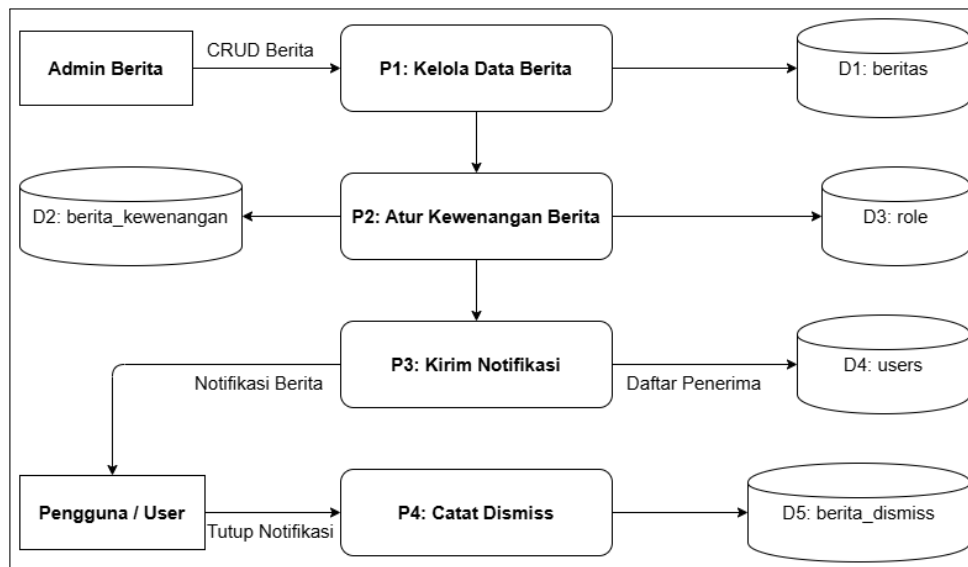


Gambar 3.2. DFD Level 0 Modul Admin Berita

### 2. DFD Level 1

Gambar 3.3 menampilkan *Data Flow Diagram* (DFD) Level 1 yang menjabarkan proses

internal pada sistem Modul Admin Berita secara lebih rinci. Pada level ini, proses utama pada sistem diuraikan menjadi empat proses inti yang saling terhubung, yaitu Kelola Data Berita, Atur Kewenangan Berita, Kirim Notifikasi, dan Catat *Dismiss*. Proses Kelola Data Berita (P1) berfungsi untuk menerima input dari Admin Berita berupa penambahan, pengubahan, maupun penghapusan data berita. Data yang telah melalui proses validasi kemudian disimpan ke dalam Data Store D1: beritas. Selanjutnya, data berita yang tersimpan digunakan oleh proses Atur Kewenangan Berita (P2) untuk mengelola keterkaitan antara berita dan peran pengguna berdasarkan data peran yang tersimpan pada Data Store D3: role, dengan hasil pengaturan kewenangan disimpan pada Data Store D2: berita\_kewenangan. Proses berikutnya adalah Kirim Notifikasi (P3), yang bertugas mengirimkan daftar berita aktif kepada pengguna sesuai dengan kewenangan yang telah ditentukan. Proses ini memanfaatkan data pengguna yang tersimpan pada Data Store D4: users dan menghasilkan notifikasi berita yang diterima oleh pengguna. Ketika pengguna melakukan penutupan (*dismiss*) terhadap notifikasi berita yang diterima, sistem akan memproses tindakan tersebut melalui proses Catat Dismiss (P4). Proses ini mencatat interaksi pengguna dan menyimpannya sebagai log pada Data Store D5: berita\_dismiss. Dengan demikian, DFD Level 1 memberikan gambaran detail mengenai alur data, pemrosesan internal, serta keterkaitan antar proses dan basis data dalam Modul Admin Berita.

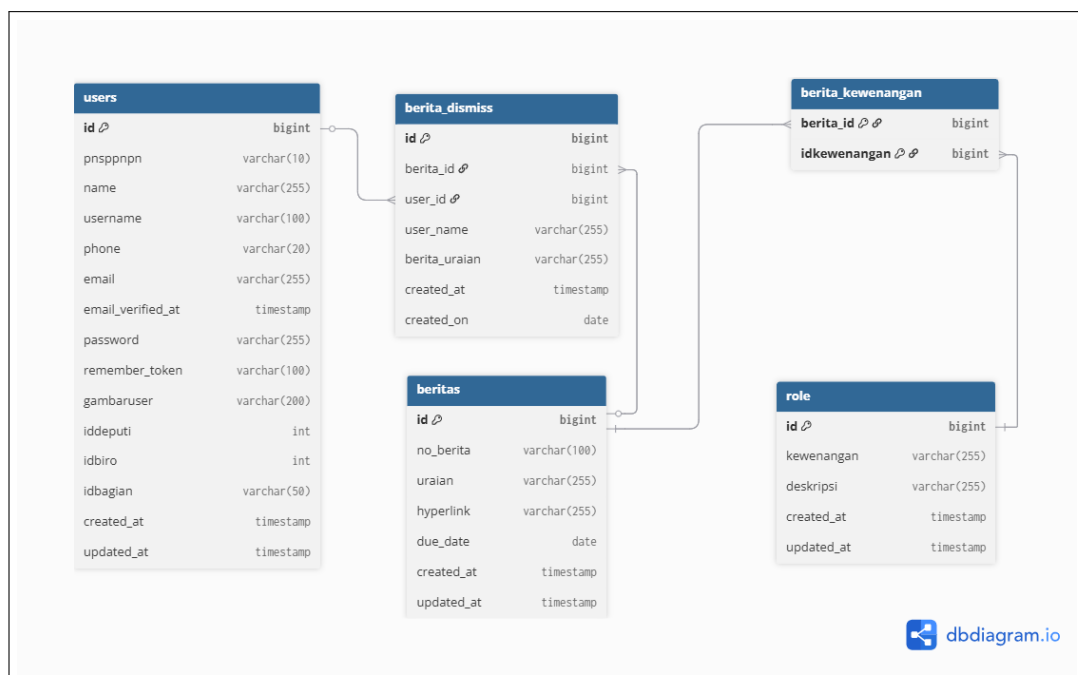


Gambar 3.3. DFD Level 1 Modul Admin Berita

## B Entity Relationship Diagram (ERD) Modul Admin Berita

Entity Relationship Diagram (ERD) pada Modul Admin Berita digunakan untuk menggambarkan struktur basis data serta hubungan antar entitas yang mendukung pengelolaan data berita dan interaksi pengguna. ERD ini berfungsi sebagai acuan untuk memahami keterkaitan data antar tabel, mulai dari penyimpanan berita, pengaturan kewenangan penerima, hingga pencatatan interaksi pengguna terhadap notifikasi berita. Gambar 3.4 menampilkan

ERD Modul Admin Berita yang terdiri atas lima tabel utama, yaitu users, beritas, berita\_kewenangan, role, dan berita\_dismiss. Hubungan antar tabel pada ERD ini menunjukkan bagaimana data berita dikaitkan dengan peran pengguna serta bagaimana interaksi pengguna terhadap notifikasi berita dicatat oleh sistem. Tabel users menyimpan data akun pengguna yang terdaftar dalam sistem. Setiap pengguna memiliki satu peran utama yang direpresentasikan melalui tabel role, sehingga hubungan antara role dan users bersifat satu ke banyak (one-to-many), di mana satu peran dapat dimiliki oleh banyak pengguna. Sebaliknya, setiap pengguna hanya terhubung dengan satu peran dalam sistem. Tabel beritas berfungsi sebagai penyimpanan utama seluruh data berita yang dikelola oleh admin dan ditampilkan kepada pengguna. Hubungan antara beritas dan berita\_kewenangan bersifat satu ke banyak, di mana satu berita dapat memiliki beberapa data kewenangan penerima. Tabel berita\_kewenangan berperan sebagai tabel penghubung antara beritas dan role, sehingga hubungan antara berita dan peran bersifat banyak ke banyak (many-to-many). Melalui relasi ini, satu berita dapat ditujukan kepada lebih dari satu peran, dan satu peran dapat menerima lebih dari satu berita. Pencatatan interaksi pengguna terhadap notifikasi berita direpresentasikan melalui tabel berita\_dismiss. Relasi antara users dan berita\_dismiss bersifat satu ke banyak, karena satu pengguna dapat menutup beberapa notifikasi berita. Selain itu, relasi antara beritas dan berita\_dismiss juga bersifat satu ke banyak, yang menunjukkan bahwa satu berita dapat ditutup oleh banyak pengguna. Dengan demikian, tabel berita\_dismiss berfungsi untuk menyimpan riwayat interaksi pengguna terhadap setiap berita yang ditampilkan oleh sistem. Dengan adanya ERD seperti yang ditunjukkan pada Gambar 3.4, relasi antar entitas pada Modul Admin Berita dapat dipahami secara menyeluruh. ERD ini menjadi dasar perancangan basis data agar proses pengelolaan berita, pengaturan kewenangan penerima, serta pelacakan interaksi pengguna dapat berjalan secara terstruktur, konsisten, dan sesuai dengan kebutuhan sistem.



Gambar 3.4. Entity Relationship Diagram (ERD) Modul Admin Berita

## C Struktur Basis Data

Struktur basis data berfungsi untuk menggambarkan rancangan logis dari sistem yang menjelaskan bagaimana data disimpan, dikelola, dan dimanfaatkan untuk mendukung proses pengelolaan berita pada Modul Admin Berita. Pembahasan pada bagian ini difokuskan pada tabel yang digunakan, fungsi masing-masing tabel, contoh kolom penting yang disimpan, serta peran tabel tersebut dalam alur proses sistem. Penjelasan mengenai relasi antar tabel tidak diulang pada bagian ini karena telah dibahas pada bagian Entity Relationship Diagram (ERD). Struktur basis data pada Modul Admin Berita terdiri atas lima tabel utama, yaitu users, beritas, berita\_dismiss, berita\_kewenangan, dan role. Kelima tabel tersebut saling melengkapi dalam mendukung proses pengelolaan konten berita, pengaturan kewenangan penerima, serta pencatatan interaksi pengguna terhadap notifikasi berita. Penjelasan masing-masing tabel disajikan sebagai berikut.

### 1. Tabel users

Tabel 3.2 digunakan untuk menyimpan data akun pengguna yang memiliki akses ke sistem. Kolom username dan password digunakan dalam proses autentikasi, kolom name dan email menyimpan identitas pengguna, sedangkan kolom iddeputi, idbiro, dan idbagian digunakan untuk pengelompokan pengguna berdasarkan struktur organisasi. Dalam proses sistem, tabel ini berperan sebagai referensi utama identitas pengguna, baik dalam penayangan berita maupun pencatatan interaksi pengguna terhadap notifikasi.

Tabel 3.2. Struktur tabel users berita

Kolom	Tipe Data	Keterangan
id	BIGINT	ID pengguna (auto increment) – Primary Key.
pnsppnpn	VARCHAR(10)	NIP atau identitas internal pegawai.
name	VARCHAR(255)	Nama lengkap pengguna.
username	VARCHAR(100)	Nama pengguna untuk login.
phone	VARCHAR(20)	Nomor telepon pengguna.
email	VARCHAR(255)	Alamat surel pengguna.
email_verified_at	TIMESTAMP	Waktu verifikasi email.
password	VARCHAR(255)	Hash kata sandi untuk autentikasi.
remember_token	VARCHAR(100)	Token fitur remember me.
gambaruser	VARCHAR(200)	Path foto profil pengguna.
iddeputi	INT	ID Deputi.
idbiro	INT	ID Biro.
idbagian	VARCHAR(50)	ID Bagian.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

### 2. Tabel beritas

Tabel 3.3 digunakan sebagai penyimpanan utama data berita. Kolom no\_berita menyimpan

identitas unik berita, kolom uraian menyimpan judul atau ringkasan berita, kolom hyperlink menyimpan tautan referensi, dan kolom due\_date menentukan batas waktu penayangan berita. Tabel ini menjadi sumber utama dalam penayangan berita pada dashboard pengguna serta acuan dalam pengaturan kewenangan dan pencatatan interaksi pengguna.

Tabel 3.3. Struktur tabel beritas

Kolom	Tipe Data	Keterangan
id	BIGINT	ID berita – Primary Key.
no_berita	VARCHAR(100)	Nomor unik berita (BR-YYMMDD-).
uraian	VARCHAR(255)	Judul atau uraian singkat berita.
hyperlink	VARCHAR(255)	Tautan referensi berita.
due_date	DATE	Batas penayangan berita.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

### 3. Tabel berita\_dismiss

Tabel 3.4 digunakan untuk mencatat interaksi pengguna terhadap notifikasi berita. Kolom berita\_id dan user\_id menunjukkan berita dan pengguna yang terlibat, sedangkan kolom created\_at dan created\_on menyimpan waktu terjadinya interaksi. Data pada tabel ini digunakan untuk menentukan status notifikasi dan menampilkan informasi pengguna yang telah menutup suatu berita.

Tabel 3.4. Struktur tabel berita\_dismiss

Kolom	Tipe Data	Keterangan
id	BIGINT	ID log penutupan notifikasi.
berita_id	BIGINT	Referensi ke tabel beritas.
user_id	BIGINT	Referensi ke tabel users.
user_name	VARCHAR(255)	Nama pengguna.
berita_uraian	VARCHAR(255)	Uraian berita.
created_at	TIMESTAMP	Waktu kejadian.
created_on	DATE	Tanggal kalender rekap.

### 4. Tabel berita\_kewenangan

Tabel 3.5 digunakan untuk menyimpan pengaturan kewenangan penerima berita. Kolom berita\_id dan idkewenangan digunakan untuk menentukan berita dan peran pengguna yang berhak melihat berita tersebut.

Tabel 3.5. Struktur tabel berita\_kewenangan

Kolom	Tipe Data	Keterangan
berita_id	BIGINT	Referensi ke tabel beritas.
idkewenangan	BIGINT	Referensi ke tabel role.

##### 5. Tabel role

Tabel 3.6 menyimpan daftar peran atau kewenangan pengguna. Kolom kewenangan menyimpan nama peran, sedangkan kolom deskripsi berisi penjelasan fungsi peran tersebut. Tabel ini menjadi dasar pengaturan hak akses dan penentuan berita yang diterima pengguna.

Tabel 3.6. Struktur tabel role

Kolom	Tipe Data	Keterangan
id	BIGINT	ID peran – Primary Key.
kewenangan	VARCHAR(255)	Nama peran pengguna.
deskripsi	VARCHAR(255)	Deskripsi singkat peran.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan terakhir.

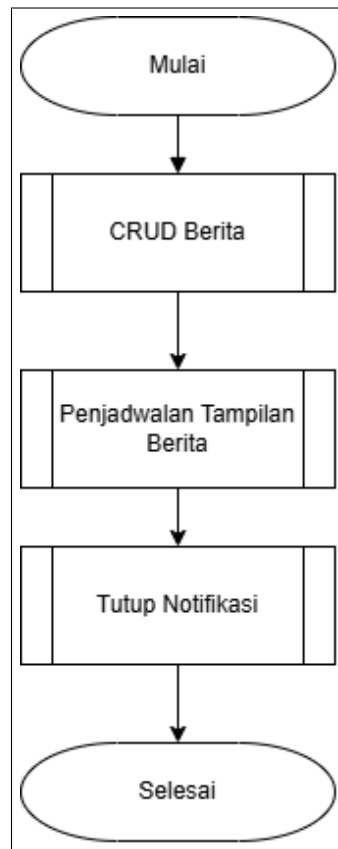
#### D Alur Proses Modul Admin Berita

Alur kerja pada Modul Admin Berita mencakup pengelolaan data berita oleh admin, penentuan penayangan berita berdasarkan tanggal kedaluwarsa, serta pencatatan interaksi pengguna saat menutup notifikasi. Untuk memudahkan pemahaman terhadap rangkaian proses tersebut, digunakan *flowchart* sebagai media visual yang menggambarkan urutan aktivitas dan titik pengambilan keputusan dalam modul ini[10]. Melalui *flowchart*, alur proses dapat ditelusuri secara lebih jelas sehingga keterkaitan antar aktivitas dapat dipahami dengan baik. Setiap bagian proses dijelaskan melalui *flowchart* berikut.

##### 1. Alur umum proses Modul Admin Berita

Gambar 3.5 memperlihatkan alur utama proses Modul Admin Berita dari awal hingga akhir. Proses diawali dengan tindakan admin dalam mengelola data berita, seperti menambah, mengubah, atau menghapus informasi berita. Selanjutnya, aplikasi melakukan penilaian terhadap tanggal kedaluwarsa berita dengan membandingkan tanggal saat ini dengan nilai *due\_date*. Berdasarkan hasil penilaian tersebut, ditentukan cara penayangan berita. Berita yang telah mendekati batas waktu penayangan akan ditampilkan sebagai notifikasi *pop-up*, sedangkan berita lainnya disajikan melalui *slider* pada halaman *dashboard*. Alur ini berakhir ketika pengguna melakukan aksi menutup notifikasi, yang menandai selesainya satu siklus penayangan berita.

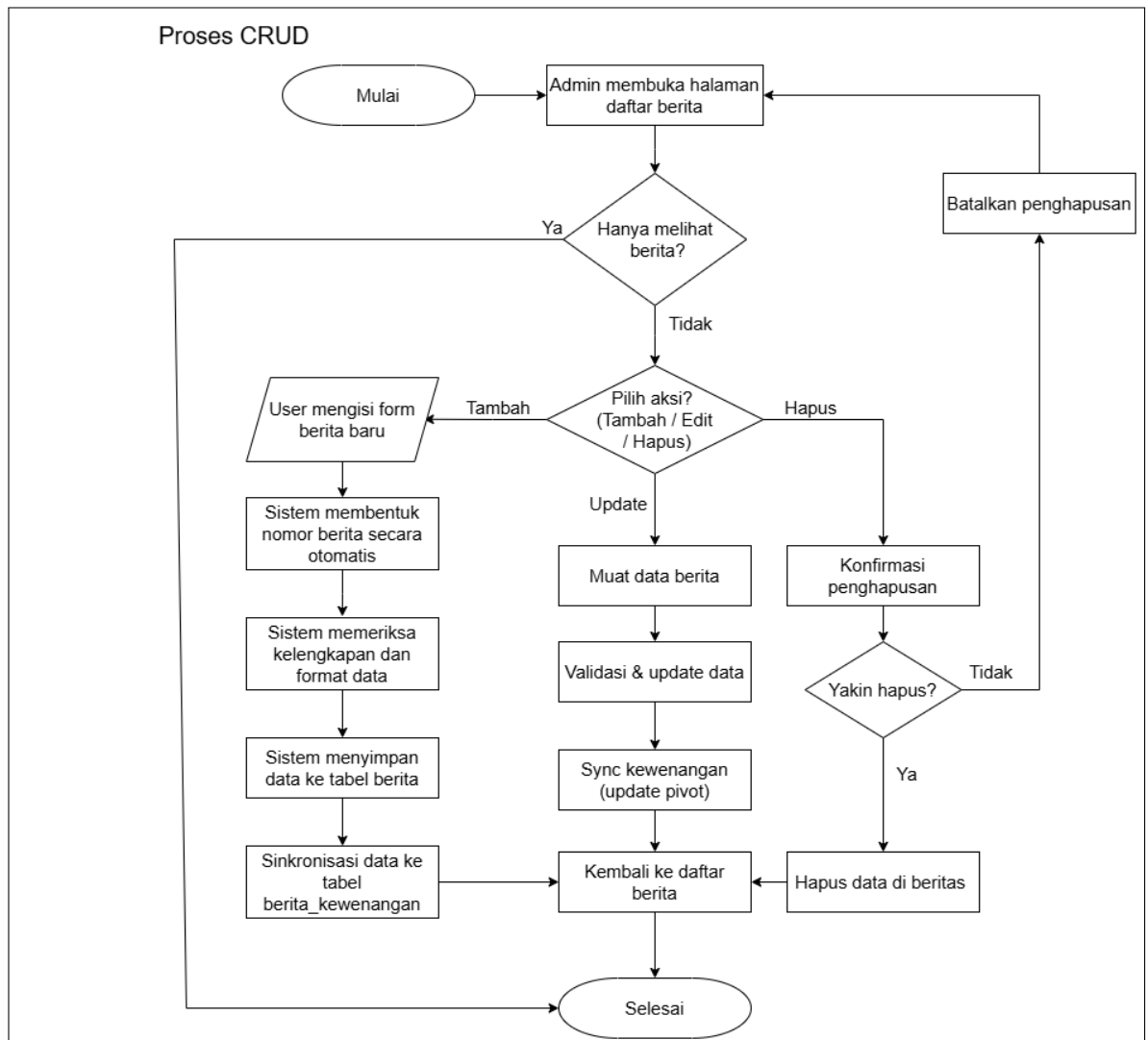




Gambar 3.5. Flowchart alur umum Modul Admin Berita

## 2. Proses pengelolaan data berita (CRUD)

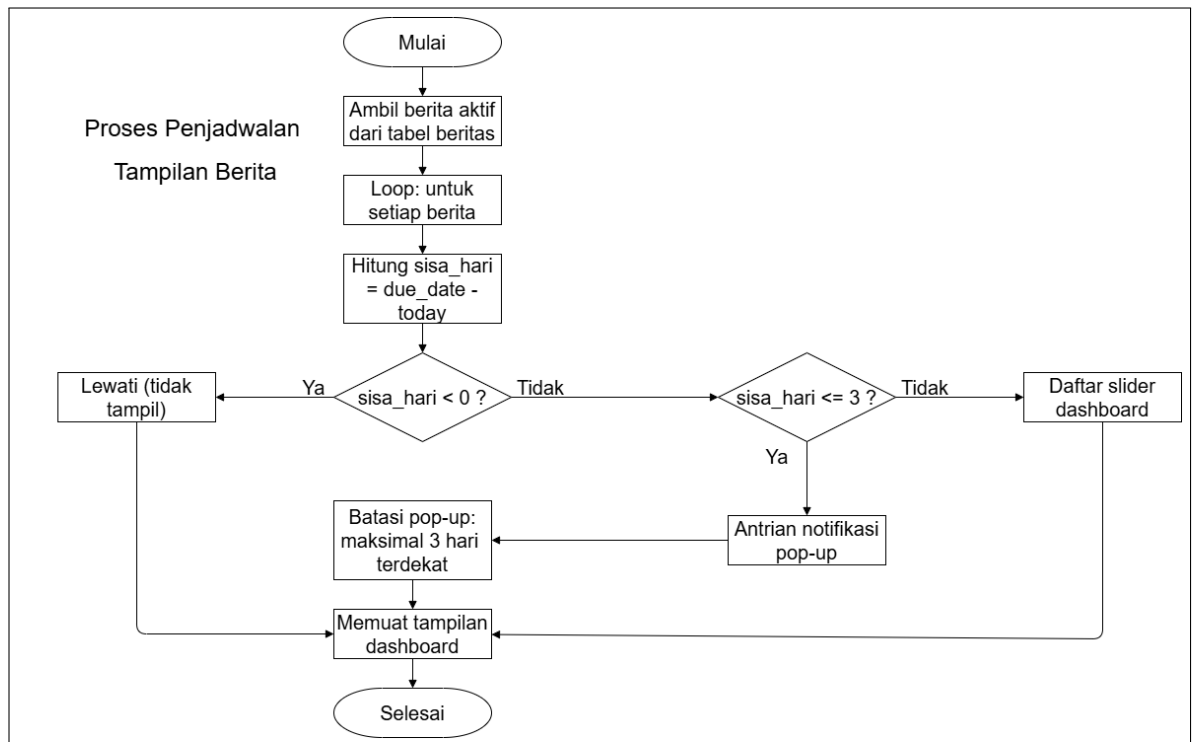
Gambar 3.6 menggambarkan tahapan pengelolaan data berita yang dilakukan oleh admin, meliputi proses tambah, ubah, dan hapus. Pada proses tambah, admin mengisi informasi berita, kemudian aplikasi melakukan pemeriksaan kelengkapan dan validitas data sebelum proses dilanjutkan. Pada proses ubah, data berita yang telah ada ditampilkan kembali untuk ditinjau, lalu dilakukan validasi sebelum perubahan disimpan. Pada proses hapus, ditampilkan permintaan konfirmasi kepada admin. Apabila konfirmasi disetujui, proses penghapusan dilanjutkan hingga data berita dihapus dan alur proses berakhir.



Gambar 3.6. Flowchart proses pengelolaan data berita (CRUD)

### 3. Proses penjadwalan dan penayangan berita

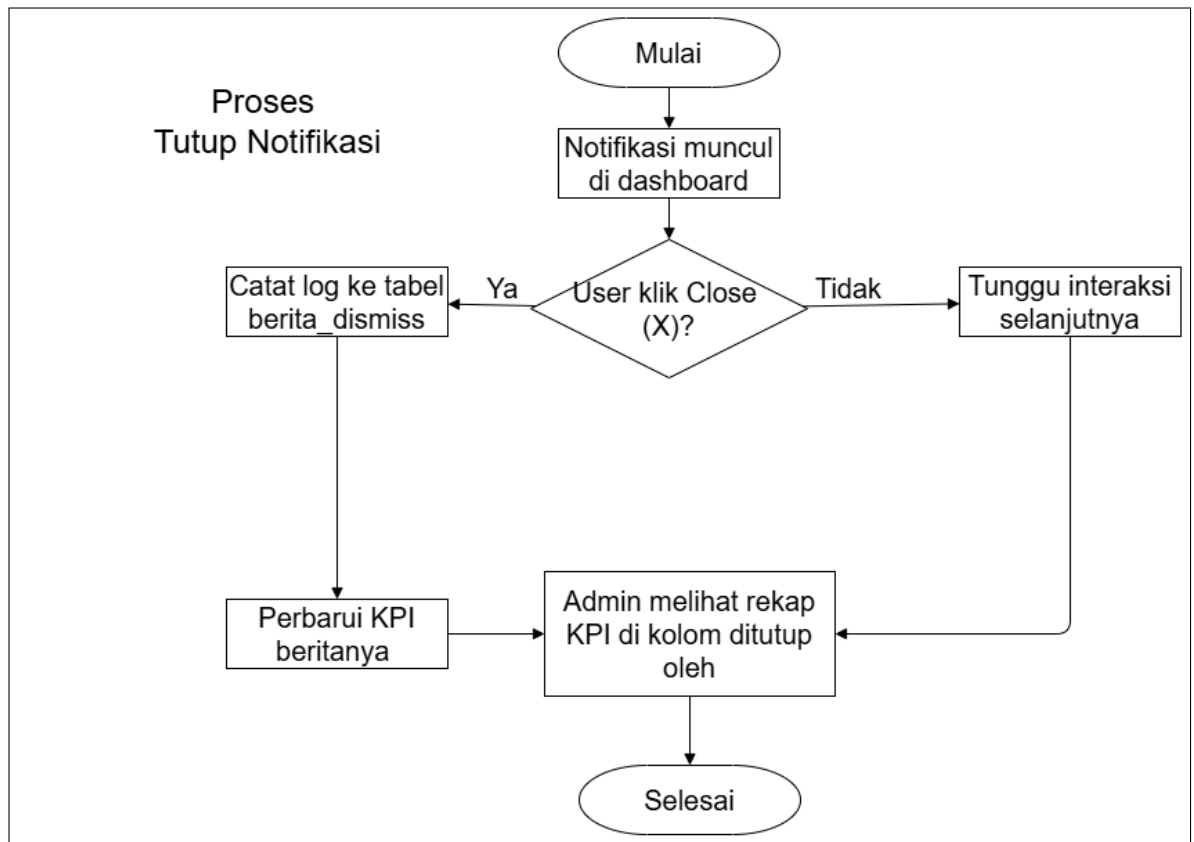
Gambar 3.7 menunjukkan mekanisme penentuan penayangan berita berdasarkan kondisi waktu. Proses dimulai dengan pengecekan status aktif berita, kemudian dilakukan perhitungan sisa waktu penayangan berdasarkan nilai *due\_date*. Apabila batas waktu penayangan telah terlewati, berita tidak lagi ditampilkan. Jika berita masih aktif dan mendekati batas waktu, berita tersebut ditampilkan sebagai notifikasi *pop-up*. Berita yang tidak memenuhi kondisi tersebut ditampilkan pada *slider* halaman *dashboard*. Dengan mekanisme ini, informasi yang bersifat mendesak dapat memperoleh prioritas penayangan.



Gambar 3.7. Flowchart proses penjadwalan dan penayangan berita

#### 4. Proses penutupan notifikasi dan pembaruan KPI

Gambar 3.8 memperlihatkan alur proses ketika pengguna menutup notifikasi berita. Proses dimulai saat pengguna menekan tombol penutup pada notifikasi. Tindakan ini diproses sebagai penanda bahwa notifikasi telah diterima atau diabaikan oleh pengguna. Selanjutnya, dilakukan pembaruan informasi kinerja dengan menghitung jumlah penutupan notifikasi serta memperbarui indikator KPI. Informasi ini digunakan sebagai bahan pemantauan dan evaluasi efektivitas penyampaian berita melalui *dashboard* admin.



Gambar 3.8. Flowchart proses penutupan notifikasi dan pembaruan KPI

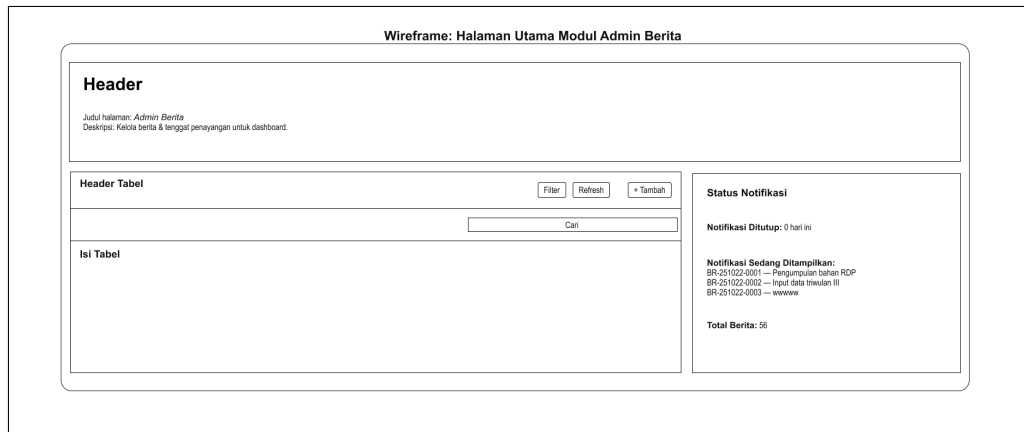
## E Wireframe Modul Admin Berita

*Wireframe* digunakan untuk memetakan tata letak antarmuka sebelum tahap implementasi dilakukan. Pada modul Admin Berita, terdapat dua tampilan utama, yaitu halaman utama admin untuk pengelolaan berita, dan halaman *dashboard* pengguna yang menampilkan notifikasi berita menjelang tenggat.

### 1. Halaman Utama Admin Berita

Gambar 3.9 menampilkan *wireframe* halaman utama Admin Berita yang digunakan sebagai acuan perancangan antarmuka sistem. *Wireframe* ini menggambarkan susunan komponen utama yang mendukung pengelolaan data berita, mulai dari proses penelusuran data hingga pengaturan notifikasi. Melalui halaman ini, admin dapat melihat daftar berita, melakukan pencarian dan pemfilteran data, serta melakukan penambahan, pengubahan, dan penghapusan berita. Selain itu, halaman ini juga menyediakan panel ringkasan status notifikasi di sisi kanan untuk membantu admin memantau kondisi penyampaian informasi kepada pengguna. Secara umum, komponen utama yang dirancang pada halaman ini meliputi header halaman yang menampilkan judul “Admin Berita” beserta deskripsi singkat fungsi halaman, bilah aksi yang memuat tombol Tambah, Filter, dan Refresh serta kolom pencarian untuk mempermudah penelusuran data, tabel berita yang menyajikan informasi utama seperti nomor berita, uraian, tautan,

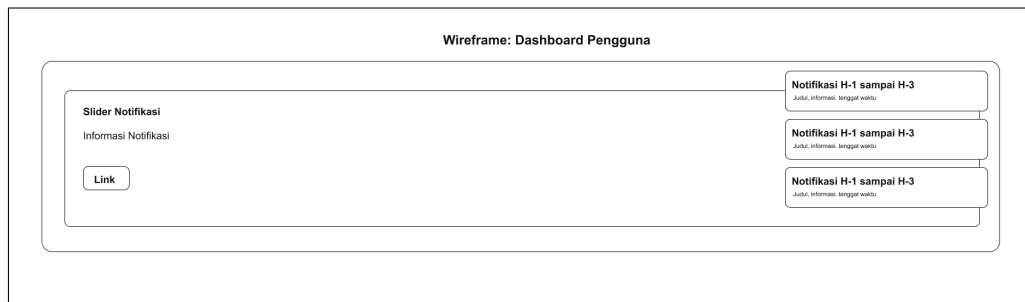
kelompok penerima, tanggal target, status penutupan, sisa hari, dan aksi pengelolaan data, serta panel status notifikasi di sisi kanan halaman. Panel tersebut menampilkan jumlah notifikasi yang sedang aktif, daftar tiga notifikasi teratas, legenda status waktu seperti H-3, H-5, dan terlewat, serta total keseluruhan berita. Dengan adanya *wireframe* ini, diharapkan rancangan antarmuka halaman utama Admin Berita dapat menjadi dasar implementasi yang konsisten dan mudah digunakan.



Gambar 3.9. *Wireframe* halaman utama Admin Berita

## 2. Dashboard Pengguna

Gambar 3.10 menampilkan *wireframe dashboard* pengguna yang digunakan sebagai acuan perancangan tampilan penayangan berita pada sistem. *Wireframe* ini menggambarkan penyajian informasi terbaru kepada pengguna sesuai dengan kewenangan yang dimiliki. Penayangan informasi dibagi menjadi dua bagian utama, yaitu *slider* informasi umum dan kartu notifikasi yang muncul secara otomatis untuk berita dengan sisa waktu H-3 hingga H-1. Pembagian ini bertujuan agar informasi penting dapat segera diketahui oleh pengguna. Komponen utama pada dashboard pengguna meliputi area *slider* informasi yang berfungsi sebagai banner untuk menampilkan satu berita dalam satu waktu dan dilengkapi dengan tombol menuju tautan berita terkait, kartu notifikasi H-3 sampai H-1 yang berisi judul, ringkasan informasi, serta tenggat waktu, serta tombol aksi pengguna untuk menutup (*dismiss*) notifikasi. Setiap tindakan penutupan notifikasi tersebut dicatat oleh sistem sebagai bentuk interaksi pengguna.



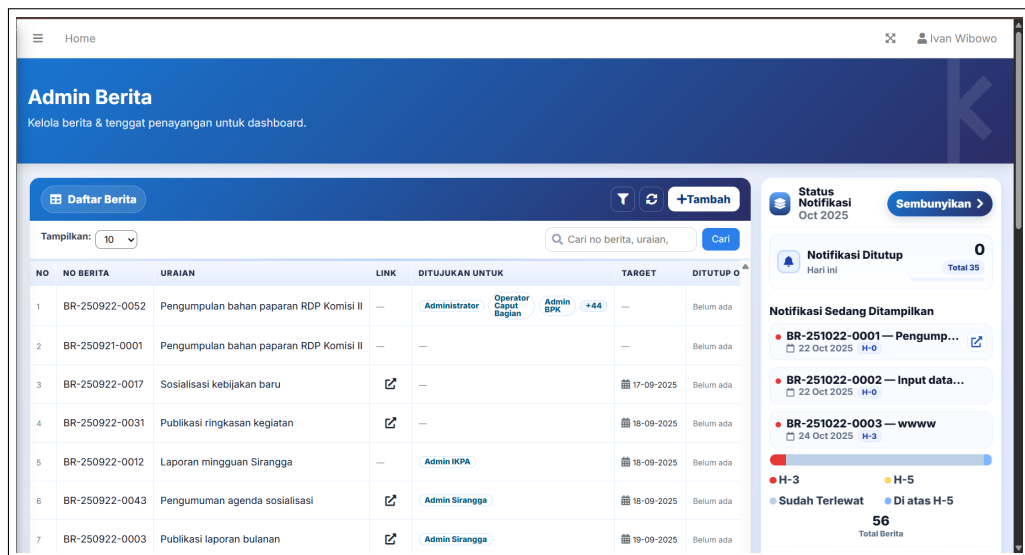
Gambar 3.10. Wireframe dashboard pengguna (penayangan berita)

## F Implementasi Antarmuka (Frontend)

Antarmuka pengguna *frontend* pada Modul Admin Berita dirancang agar sederhana, informatif, dan konsisten dengan tema visual DIGITall Keuangan. Implementasi dilakukan menggunakan *Blade Template* pada Laravel dengan dukungan komponen DataTables dan ikon Font Awesome.

### 1. Halaman Utama Admin Berita

Halaman utama modul menampilkan daftar berita dalam bentuk tabel yang mudah dibaca, disertai fitur pencarian, pemfilteran, serta tombol untuk menambah, mengubah, dan menghapus berita seperti ditunjukkan pada Gambar 3.11. Elemen visual seperti gradasi warna dan ikon membantu pengguna memahami hierarki informasi, sedangkan panel di sisi kanan menampilkan ringkasan notifikasi yang sedang aktif.



Gambar 3.11. Halaman utama Modul Admin Berita

Bagian antarmuka halaman utama terdiri atas beberapa komponen berikut.

- Bagian atas (header)

Bagian ini menampilkan judul “Admin Berita” beserta deskripsi “Kelola berita dan

tenggat penayangan untuk dashboard.” Latar belakang menggunakan gradasi warna biru (#186FC4 hingga #A9D3F5) yang menjadi identitas visual DIGITall. Logo lembaga berwarna putih dengan tingkat transparansi rendah ditempatkan di sisi kanan untuk memperkuat kesan resmi.

- Tabel daftar berita

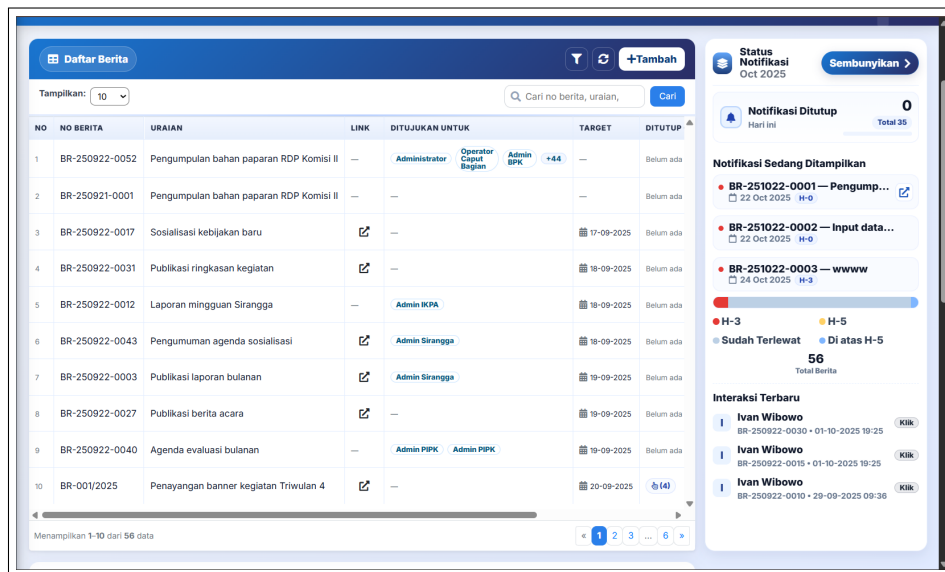
Bagian utama halaman berisi tabel dinamis yang menampilkan data berita seperti terlihat pada Gambar 3.12. Setiap baris tabel memuat nomor berita, uraian, tautan, kelompok penerima, tanggal target, status penutupan, sisa hari, dan tombol aksi. Tabel ini dibangun menggunakan DataTables sehingga mendukung fitur *sorting*, *pagination*, dan *real-time search*.

NO	NO BERITA	URAIAN	LINK	DITUJUKAN UNTUK	TARGET	DITUTUP OLEH	SISA HARI	AKSI
1	BR-250922-0052	Pengumpulan bahan paparan RDP Komisi II	—	Administrator Operator Capat Bagian Admin BPK +44	—	Belum ada	—	
2	BR-250921-0001	Pengumpulan bahan paparan RDP Komisi II	—	—	—	Belum ada	—	
3	BR-250922-0017	Sosialisasi kebijakan baru		—	17-09-2025	Belum ada	X Lewat 35 hari	
4	BR-250922-0031	Publikasi ringkasan kegiatan		—	18-09-2025	Belum ada	X Lewat 34 hari	
5	BR-250922-0012	Laporan mingguan Sirangga	—	Admin IKPA	18-09-2025	Belum ada	X Lewat 34 hari	
6	BR-250922-0043	Pengumuman agenda sosialisasi		Admin Sirangga	18-09-2025	Belum ada	X Lewat 34 hari	
7	BR-250922-0003	Publikasi laporan bulanan		Admin Sirangga	19-09-2025	Belum ada	X Lewat 33 hari	
8	BR-250922-0027	Publikasi berita acara		—	19-09-2025	Belum ada	X Lewat 33 hari	
9	BR-250922-0040	Agenda evaluasi bulanan	—	Admin PIPK Admin PIPK	19-09-2025	Belum ada	X Lewat 33 hari	
10	BR-001/2025	Penayangan banner kegiatan Triwulan 4		—	20-09-2025	(4)	X Lewat 32 hari	

Gambar 3.12. Tabel daftar berita dengan fitur pencarian dan aksi

- Panel status notifikasi

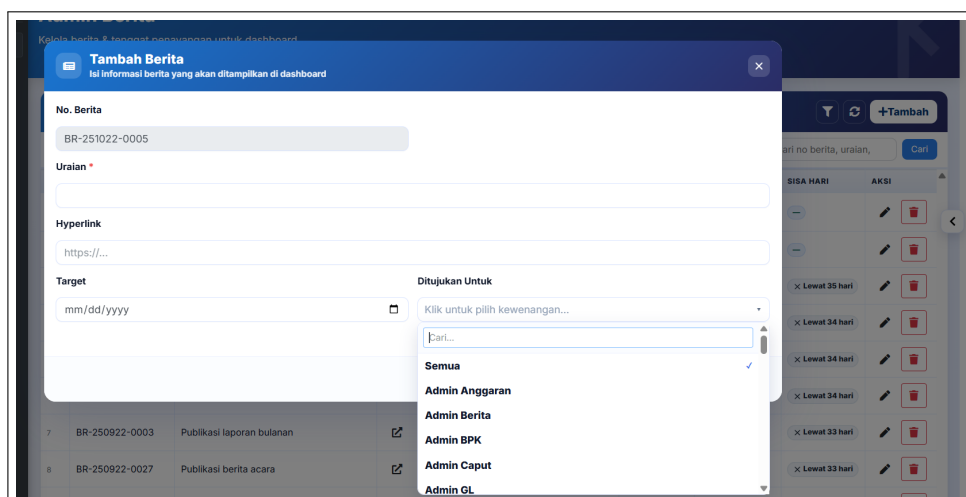
Panel di sisi kanan halaman seperti pada Gambar 3.13 menampilkan rekapitulasi status notifikasi berdasarkan kategori, yaitu berita yang sudah ditutup, notifikasi yang sedang aktif, serta jumlah total berita. Panel ini terhubung langsung dengan data dari tabel berita\_dismiss sehingga perubahan data dapat diperbarui secara otomatis.



Gambar 3.13. Panel status notifikasi aktif pada Modul Admin Berita

- Formulir tambah dan ubah berita

Sistem menampilkan jendela *popup form* ketika admin melakukan penambahan atau pengubahan data berita, seperti ditunjukkan pada Gambar 3.14 dan Gambar 3.15. Formulir ini mencakup beberapa elemen utama, yaitu nomor berita yang dibuatkan secara otomatis, uraian berita, tautan terkait, tanggal target penayangan, serta pilihan kewenangan penerima berita. Pilihan kewenangan disediakan dalam bentuk *multi-select dropdown* sehingga admin dapat menetapkan lebih dari satu kelompok pengguna sebagai penerima informasi dalam satu berita.



Gambar 3.14. Formulir tambah berita



Gambar 3.15. Formulir ubah berita dengan pilihan kewenangan

- Tooltip interaksi pengguna

Pada kolom “Ditutup oleh” seperti ditunjukkan pada Gambar 3.16, sistem menampilkan ikon yang menunjukkan jumlah pengguna yang telah menutup notifikasi suatu berita. Ketika ikon diarahkan menggunakan kursor, sistem akan menampilkan *tooltip* yang berisi daftar nama pengguna beserta jumlah interaksi yang dilakukan. Fitur ini memudahkan admin dalam memantau tingkat keterlibatan pengguna terhadap notifikasi berita secara langsung tanpa perlu membuka halaman laporan terpisah.

7	BR-250922-0003	Publikasi laporan bulanan		Admin Sirangga	19-09-2025	Belum ada	X Lewat 33 hari		
8	BR-250922-0027	Publikasi berita acara		—	19-09-2025	Belum ada	X Lewat 33 hari		
9	BR-250922-0040	Agenda evaluasi bulanan	—	Admin PIPK Admin PIPK	19-09-2025	Belum ada	X Lewat 33 hari		
10	BR-001/2025	Penayangan banner kegiatan Triwulan 4		—	20-09-2025	(4)	X Lewat 32 hari		

Ditutup oleh (total 4)

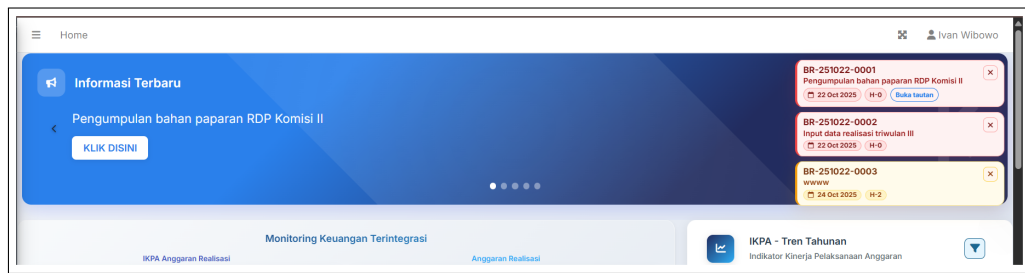
- Ivan Wibowo (3)
- Pengguna (1)

Menampilkan 1-10 dari 56 data

Gambar 3.16. Tampilan tooltip interaksi pengguna terhadap notifikasi

## 2. Tampilan pada *Dashboard* Pengguna

Setelah berita ditambahkan dan mencapai waktu penayangan (H-5 atau kurang), sistem menampilkan berita secara otomatis pada *dashboard* pengguna sesuai dengan kewenangan yang dimiliki, seperti ditunjukkan pada Gambar 3.17. Berita ditampilkan dalam bentuk notifikasi *popup* dengan tombol “Buka tautan” dan ikon penutup, serta dalam bentuk *slider* “Informasi Terbaru” di bagian atas *dashboard*. Desain ini memastikan informasi penting tetap terlihat tanpa mengganggu aktivitas utama pengguna.



Gambar 3.17. Penayangan berita pada halaman dashboard pengguna

## G Implementasi *Backend*

Bagian ini menjelaskan proses implementasi logika pada Modul Admin Berita. Seluruh mekanisme *backend* ditulis dalam bentuk pseudocode untuk mempermudah penelusuran dan pemahaman alur kerja sistem tanpa terikat pada sintaks bahasa pemrograman tertentu.

### 1. Pembangkitan Nomor Berita Otomatis.

Salah satu fitur penting dalam Modul Admin Berita adalah pembangkitan nomor berita secara otomatis. Nomor ini mengikuti pola BR-YYMMDD-, di mana urutan empat digit pada bagian akhir diatur ulang setiap pergantian hari. Proses ini diimplementasikan melalui fungsi `nextNoBerita` yang mencari entri terakhir berdasarkan tanggal saat ini, lalu menaikkan urutan angka secara berurutan dan aman. Dengan pendekatan ini, sistem dapat menjamin keunikan setiap nomor berita yang dibuat pada hari yang sama. Pseudocode proses ini dapat dilihat pada Kode 3.1.

```

1 FUNCTION generateNomorBerita(tanggalHariIni):
2     // Bentuk awalan nomor, misal: BR-251023-
3     prefix = "BR-" + formatTanggal(tanggalHariIni, "yymmdd"
4         " ") + "-"
5
6     // Cari nomor terakhir dengan awalan yang sama
7     nomorTerakhir = SELECT no_berita
8                     FROM beritas
9                     WHERE no_berita LIKE prefix + "%"
10                    ORDER BY no_berita DESC
11                    LIMIT 1
12
13    // Tentukan urutan berikutnya
14    IF nomorTerakhir ditemukan THEN
15        urutanBaru = ambilempatDigitAkhir(nomorTerakhir) +
16        1
17    ELSE
18        urutanBaru = 1
19    ENDIF

```

```

18
19 // Gabungkan prefix dengan nomor urut empat digit
20 nomorBaru = prefix + formatAngka(urutanBaru, 4)
21
22 RETURN nomorBaru
23 END FUNCTION

```

### Kode 3.1: Pseudocode pembangkitan nomor berita otomatis

Contoh penggunaan: ketika admin membuka formulir penambahan data baru, sistem akan memanggil fungsi nextNoBerita untuk mengisi nilai awal pada kolom nomor berita. Saat tombol simpan ditekan, nomor tersebut dikirim kembali untuk divalidasi, memastikan tidak ada duplikasi data.

#### 2. Pengambilan Daftar, Ringkasan, dan KPI.

Fungsi list berperan untuk menyiapkan keseluruhan data yang akan ditampilkan pada halaman utama Modul Admin Berita. Proses ini meliputi pencarian dengan kata kunci, penyaringan berdasarkan kewenangan pengguna, serta pembentukan ringkasan status berdasarkan sisa hari menuju tanggal kedaluwarsa berita. Selain itu, fungsi ini juga menghitung metrik kinerja atau KPI terkait jumlah notifikasi yang telah ditutup. Data hasil pengolahan kemudian digunakan untuk mengisi tabel daftar berita, menampilkan ringkasan status, serta menampilkan tiga notifikasi terdekat (H0–H3) di antarmuka pengguna. Pseudocode-nya ditunjukkan pada Kode 3.2.

```

1 FUNCTION tampilkanDaftarBerita(parameter):
2     // Ambil semua data berita beserta kewenangannya
3     dataBerita = SELECT * FROM beritas JOIN
4         berita_kewenangan
5
6     // Jika ada kata kunci pencarian
7     IF parameter.cari TIDAK KOSONG THEN
8         dataBerita = dataBerita WHERE
9             no_berita    MENGANDUNG parameter.cari
10            OR uraian     MENGANDUNG parameter.cari
11            OR hyperlink  MENGANDUNG parameter.cari
12     ENDIF
13
14     // Jika pengguna memilih filter kewenangan tertentu
15     IF parameter.idKewenangan TIDAK KOSONG THEN
16         dataBerita = dataBerita WHERE
17             berita.id ADA DI (
18                 SELECT berita_id FROM
19                     berita_kewenangan
20                     WHERE idkewenangan DALAM parameter
21                     .idKewenangan
22             )

```

```

20     ENDIF
21
22     // Urutkan berdasarkan tanggal kedaluwarsa
23     daftarBerita = dataBerita ORDER BY due_date ASC
24
25     // Hitung ringkasan jumlah berita berdasarkan level
26     status
27     ringkasan = {
28         bahaya      : HITUNG(daftarBerita DIMANA level == "
29         danger"),
30         peringatan: HITUNG(daftarBerita DIMANA level == "
31         warning"),
32         lewatBatas: HITUNG(daftarBerita DIMANA level == "
33         overdue")
34     }
35
36     // Ambil berita yang akan kedaluwarsa dalam 3 hari ke
37     depan
38     daftarNotifikasi = SELECT * FROM beritas
39                             WHERE due_date ANTARA hariIni() DAN
40                             hariIni()+3
41                             ORDER BY due_date ASC
42
43     notifikasiAktif = JUMLAH(daftarNotifikasi)
44     notifikasiDitampilkan = AMBIL(daftarNotifikasi, 3) //
45     hanya tampilkan 3 teratas
46
47     // Hitung berita yang sudah ditutup (dismiss) oleh
48     pengguna
49     ditutupHariIni = HITUNG DISTINCT(berita_id, user_id)
50                             DARI berita_dismiss
51                             DIMANA tanggal = hariIni()
52
53     ditutupTotal = HITUNG DISTINCT(berita_id, user_id)
54                             DARI berita_dismiss
55
56     // Kembalikan semua hasil dalam satu paket data
57     KEMBALIKAN {
58         daftarBerita,
59         ringkasan,
60         notifikasiDitampilkan,
61         notifikasiAktif,
62         ditutupHariIni,

```

```

55         ditutupTotal ,
56         nomorBeritaBaru : generateNomorBerita(hariIni())
57     }
58 END FUNCTION

```

### Kode 3.2: Pseudocode daftar dan ringkasan data berita

Contoh penggunaan: ketika halaman utama dimuat, controller akan memanggil fungsi list dengan parameter pencarian dan penyaringan tertentu. Hasil pemanggilan ini digunakan untuk menampilkan tabel berita, panel ringkasan level risiko, jumlah notifikasi aktif, serta nomor berita berikutnya yang siap digunakan.

#### 3. Menyimpan dan Memperbarui Berita.

Dua prosedur utama pada bagian ini adalah create dan update, yang bertugas untuk menangani proses validasi, penyimpanan, serta pembaruan data berita. Saat proses pembuatan berlangsung, sistem akan membangkitkan nomor berita baru secara otomatis melalui fungsi nextNoBerita. Selanjutnya, data berita disimpan ke dalam tabel beritas dan dilakukan sinkronisasi daftar kewenangan ke tabel berita\_kewenangan. Pada proses pembaruan, data yang diperbarui hanya mencakup uraian, tautan, tanggal jatuh tempo, dan daftar kewenangan, sementara nomor berita tetap dipertahankan. Pseudocode-nya disajikan pada Kode 3.3.

```

1 FUNCTION simpanBerita(dataInput):
2     // Periksa kelengkapan data yang dikirim
3     VALIDASI dataInput:
4         - uraian wajib diisi
5         - hyperlink harus berformat URL (jika ada)
6         - due_date harus berupa tanggal (jika ada)
7         - minimal satu kewenangan dipilih
8
9     // Buat nomor berita otomatis
10    nomorBaru = generateNomorBerita(hariIni())
11
12    // Simpan data berita ke tabel utama
13    idBaru = INSERT INTO beritas (
14        no_berita, uraian, hyperlink, due_date
15    ) VALUES (
16        nomorBaru, dataInput.uraian, dataInput.
17        hyperlink, dataInput.due_date
18    )
19
20    // Simpan kewenangan berita ke tabel relasi
21    SINKRONISASI berita_kewenangan
22    UNTUK berita_id = idBaru
23    DENGAN daftar kewenangan dari dataInput

```

```

24     RETURN idBaru
25 END FUNCTION
26
27
28 FUNCTION perbaruiBerita(idBerita, dataInput):
29     // Periksa kembali data sebelum diperbarui
30     VALIDASI dataInput:
31         - uraian wajib diisi
32         - hyperlink harus berformat URL (jika ada)
33         - due_date harus berupa tanggal (jika ada)
34         - minimal satu kewenangan dipilih
35
36     // Perbarui data berita di tabel utama
37     UPDATE beritas SET
38         uraian = dataInput.uraian,
39         hyperlink= dataInput.hyperlink,
40         due_date = dataInput.due_date
41     WHERE id = idBerita
42
43     // Perbarui daftar kewenangan berita terkait
44     SINKRONISASI berita_kewenangan
45         UNTUK berita_id = idBerita
46         DENGAN daftar kewenangan dari dataInput
47
48     RETURN "Berhasil diperbarui"
49 END FUNCTION

```

Kode 3.3: Pseudocode penyimpanan dan pembaruan data berita

Contoh penggunaan: ketika pengguna menyimpan berita baru, sistem akan memvalidasi data terlebih dahulu, membuat nomor berita otomatis, kemudian menyimpan data ke dalam basis data. Sementara itu, saat dilakukan pembaruan, sistem hanya mengubah data yang relevan tanpa mengubah identitas berita yang telah ada.

#### 4. Penentuan Tempat Notifikasi Tampil Berdasarkan Sisa Hari.

Proses ini berfungsi untuk menentukan posisi dan cara penayangan berita pada antarmuka pengguna. Logika sistem akan menghitung selisih hari antara tanggal saat ini dan kolom *due\_date*, kemudian mengelompokkan hasilnya ke dalam kategori *overdue*, *danger*, *warning*, atau *safe*. Hasil pengelompokan tersebut digunakan untuk menentukan apakah berita akan muncul pada komponen *slider* atau sebagai notifikasi *popup*. Dengan mekanisme ini, berita yang mendekati tenggat waktu akan lebih menonjol dan mudah diperhatikan oleh pengguna. Pseudocode-nya dapat dilihat pada Kode 3.4.

```

1 FUNCTION hitungSisaHari(tanggalJatuhTempo):
2     // Jika tidak ada tanggal jatuh tempo, kembalikan NULL
3     IF tanggalJatuhTempo TIDAK ADA THEN

```

```

4         RETURN NULL
5     ENDIF
6
7     // Hitung selisih hari antara hari ini dan tanggal
    jatuh tempo
8     selisih = jumlahHariAntara(hariIni(),
    tanggalJatuhTempo)
9     RETURN selisih
10 END FUNCTION
11
12
13 FUNCTION tentukanLevel(sisaHari):
14     // Jika tidak ada data tanggal, level tidak ditentukan
15     IF sisaHari TIDAK ADA THEN
16         RETURN NULL
17     ENDIF
18
19     // Tentukan kategori berdasarkan jarak hari
20     IF sisaHari < 0 THEN
21         RETURN "overdue"           // sudah lewat batas waktu
22     ELSE IF sisaHari <= 3 THEN
23         RETURN "danger"           // mendekati tenggat (H-3)
24     ELSE IF sisaHari <= 5 THEN
25         RETURN "warning"         // tenggat dekat (H-5)
26     ELSE
27         RETURN "safe"            // masih aman
28     ENDIF
29 END FUNCTION
30
31
32 FUNCTION tentukanTampilan(tanggalJatuhTempo):
33     sisa = hitungSisaHari(tanggalJatuhTempo)
34
35     // Jika tidak ada tanggal, tampilkan hanya di slider
36     IF sisa TIDAK ADA THEN
37         RETURN { slider: true, popup: false }
38     ENDIF
39
40     // Jika sudah lewat tenggat, tidak perlu ditampilkan
41     IF sisa < 0 THEN
42         RETURN { slider: false, popup: false }
43     ENDIF
44

```

```

45 // Jika akan jatuh tempo dalam 3 hari, tampilkan
    sebagai popup notifikasi
46 IF sisa <= 3 THEN
47     RETURN { slider: false, popup: true }
48 ENDIF
49
50 // Selain kondisi di atas, tampilkan di slider saja
51 RETURN { slider: true, popup: false }
52 END FUNCTION

```

Kode 3.4: Pseudocode logika tampilan berita (slider dan notifikasi)

Contoh penggunaan: ketika halaman *dashboard* ditampilkan, sistem hanya akan menampilkan berita yang masih aktif di komponen *slider*, sedangkan tiga berita dengan tenggat waktu paling dekat (H0–H3) akan muncul dalam bentuk notifikasi *popup*. Berita yang sudah melewati tenggat tidak ditampilkan kembali.

##### 5. Pencatatan *Dismiss* dan Pembaruan Sesi.

Setiap kali pengguna menutup notifikasi berita, sistem mencatat tindakan tersebut agar tidak menampilkan notifikasi yang sama dalam sesi berikutnya. Pencatatan dilakukan dengan menambahkan satu baris data baru ke tabel *berita.dismiss*, yang berisi identitas berita, pengguna, dan waktu penutupan. Selain itu, sistem juga memperbarui sesi pengguna dengan daftar ID berita yang telah ditutup. Pseudocode untuk proses ini ditunjukkan pada Kode 3.5.

```

1 FUNCTION tutupNotifikasi(idBerita, pengguna):
2     // Pastikan berita yang akan ditutup ada di sistem
3     VALIDASI bahwa idBerita TERDAFTAR di tabel beritas
4
5     // Simpan data penutupan (dismiss) ke tabel log
6     INSERT INTO berita_dismiss (
7         berita_id, user_id, user_name, berita_uraian,
8         created_at
9     )
10    SELECT
11        id, pengguna.id, pengguna.namaAtauEmail, uraian,
12        waktuSekarang()
13    FROM beritas
14    WHERE id = idBerita
15
16    // Tambahkan id berita yang ditutup ke sesi pengguna
17    session.daftarBeritaTertutup = GABUNG_UNIK(
18        session.daftarBeritaTertutup, { idBerita }
19    )
20
21    // Kembalikan hasil operasi ke sistem

```



```

20     RETURN {
21         status: "berhasil",
22         daftarTertutup: session.daftarBeritaTertutup
23     }
24 END FUNCTION

```

Kode 3.5: Pseudocode pencatatan tindakan penutupan notifikasi (dismiss)

Contoh penggunaan: ketika pengguna menekan tombol tutup pada *pop up*, sisi *frontend* akan memanggil *endpoint dismiss*. Sistem kemudian menyimpan data ke tabel *berita.dismiss* dan memperbarui daftar ID yang telah ditutup pada sesi aktif, sehingga notifikasi yang sama tidak akan muncul kembali selama sesi tersebut berlangsung.

### 3.3.4 Modul Lembur untuk Kepala Bagian

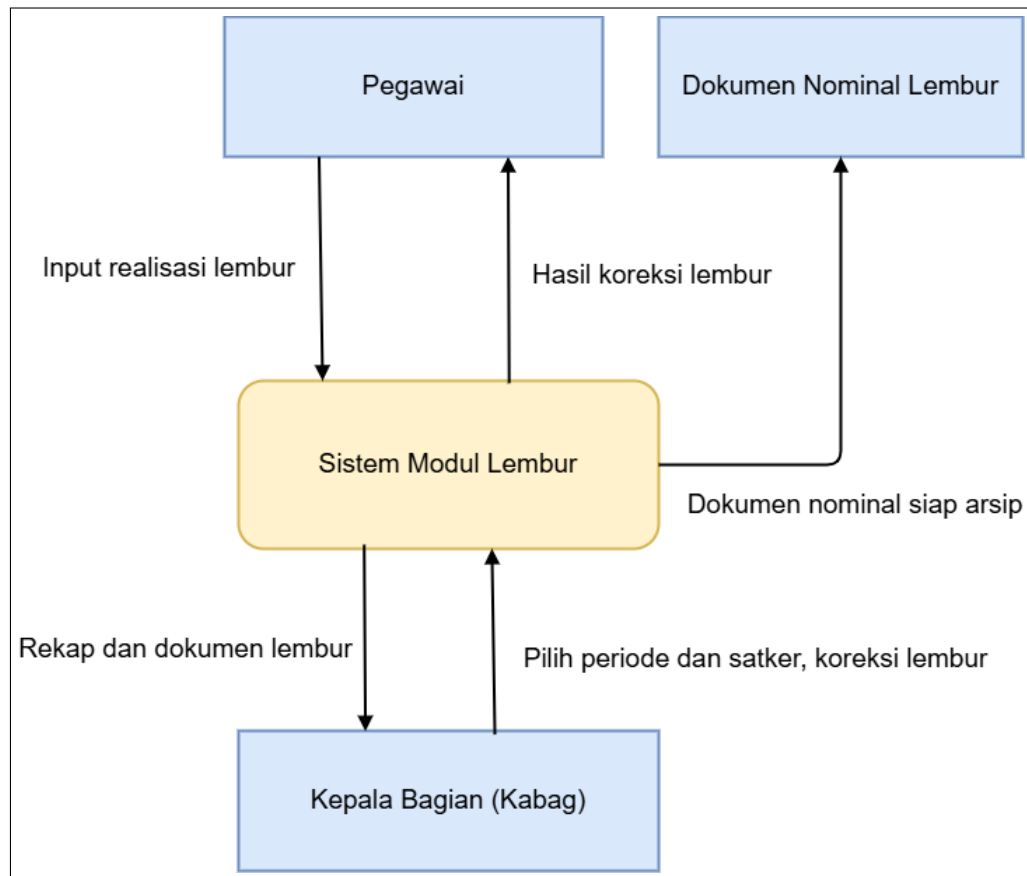
Modul Lembur pada sistem DIGITall Keuangan digunakan untuk mendukung pengelolaan administrasi lembur pegawai secara terintegrasi. Secara garis besar, modul ini mencakup proses pengajuan lembur oleh pegawai hingga penetapan nominal lembur sebagai dasar proses penandatanganan digital. Modul Lembur untuk Kepala Bagian difokuskan pada proses pemeriksaan dan validasi pengajuan lembur yang telah diinput oleh pegawai. Kepala Bagian melakukan pengecekan kesesuaian jam lembur dengan aktivitas kerja yang dilaksanakan dan memberikan koreksi apabila ditemukan ketidaksesuaian.

#### A Data Flow Diagram (DFD) Modul Lembur

Data Flow Diagram (DFD) digunakan untuk menggambarkan alur aliran data pada Modul Lembur yang berfokus pada peran Kepala Bagian (Kabag). DFD ini menunjukkan bagaimana data lembur diproses saat melakukan koreksi dan validasi data. Penyajian DFD dilakukan dalam dua tingkat, yaitu DFD Level 0 dan DFD Level 1, sebagai berikut.

##### 1. DFD Level 0

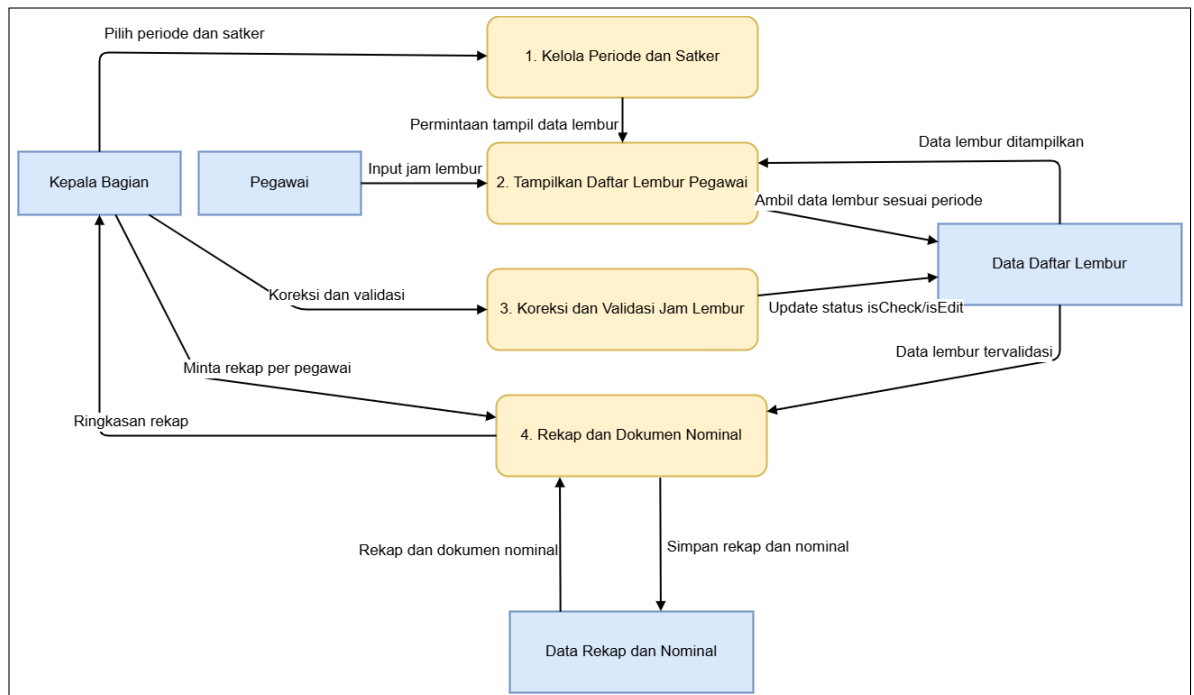
DFD Level 0 dapat dilihat pada Gambar 3.18 merupakan diagram konteks yang menggambarkan sistem Modul Lembur sebagai satu proses utama. Pada level ini, sistem berinteraksi dengan entitas luar yang terlibat langsung, yaitu Pegawai dan Kepala Bagian (Kabag). Pegawai melakukan input realisasi lembur ke dalam sistem berupa jam datang dan jam pulang. Data tersebut kemudian diproses oleh sistem untuk selanjutnya diperiksa oleh Kabag. Kepala Bagian berinteraksi dengan sistem untuk memilih periode dan satuan kerja yang berada di bawah kewenangannya, melakukan koreksi lembur, serta menerima hasil rekap dan dokumen nominal lembur. Keluaran dari sistem berupa dokumen nominal lembur yang telah siap digunakan untuk proses arsip atau administrasi lanjutan.



Gambar 3.18. Data Flow Diagram Level 0 Modul Lembur

## 2. DFD Level 1

DFD Level 1 dapat dilihat pada Gambar 3.19 merupakan pengembangan dari DFD Level 0 yang menjabarkan proses utama ke dalam beberapa proses yang lebih rinci sesuai dengan alur kerja Kepala Bagian. Proses pertama adalah pengelolaan periode dan satuan kerja, yaitu tahap ketika Kabag menentukan periode lembur dan satuan kerja yang akan diperiksa. Proses kedua adalah penampilan daftar lembur pegawai. Pada tahap ini, sistem menampilkan data lembur pegawai sesuai periode dan satuan kerja yang dipilih, berdasarkan data realisasi lembur yang telah diinput oleh pegawai. Proses ketiga adalah koreksi dan validasi jam lembur, yaitu tahap di mana Kabag memeriksa kebenaran jam lembur serta melakukan koreksi apabila ditemukan ketidaksesuaian data. Proses keempat adalah rekap dan penyusunan dokumen nominal lembur. Pada tahap ini, data lembur yang telah tervalidasi direkapitulasi per pegawai dan per satuan kerja, kemudian disusun menjadi dokumen nominal lembur. Hasil dari proses ini berupa ringkasan rekap dan dokumen nominal lembur yang dapat diakses oleh Kepala Bagian.

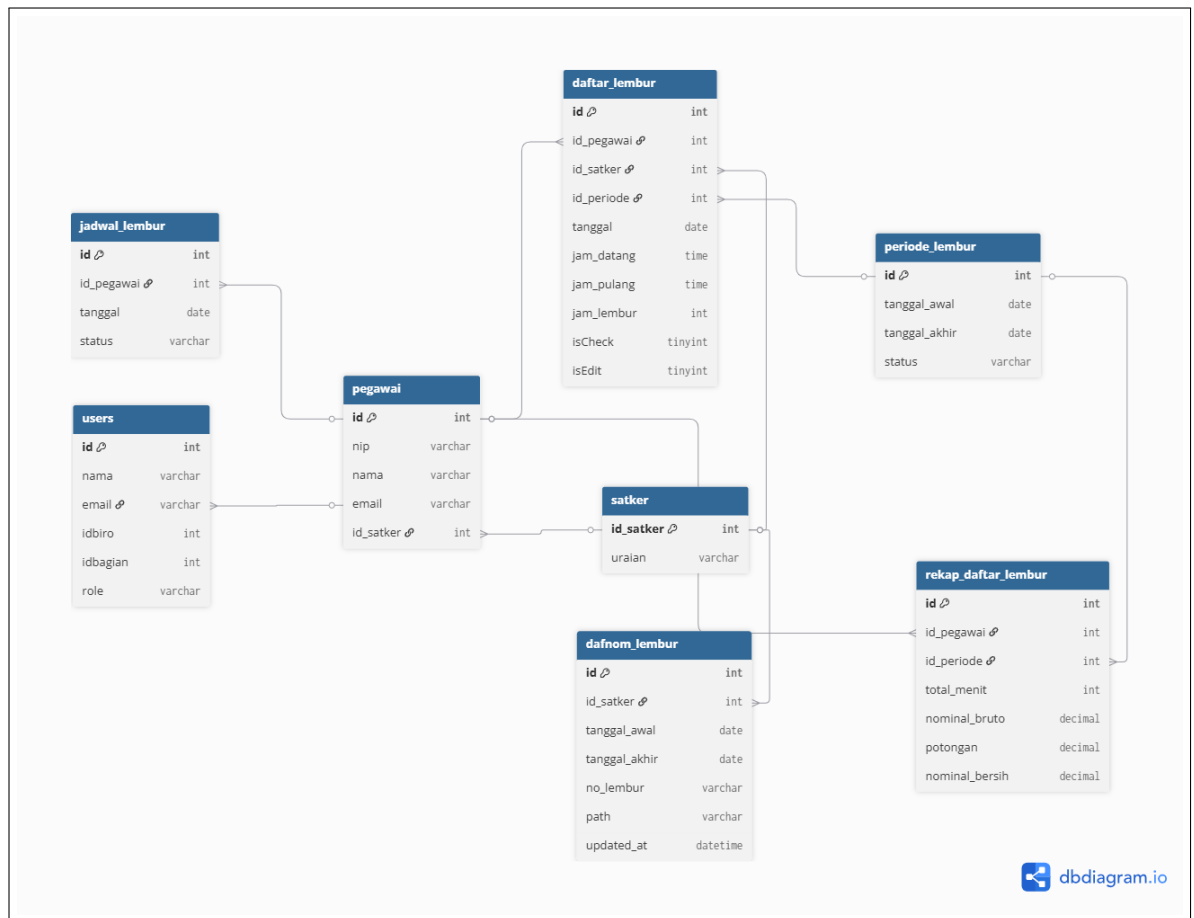


Gambar 3.19. Data Flow Diagram Level 1 Modul Lembur

## B Entity Relationship Diagram (ERD) Modul Lembur

Entity Relationship Diagram (ERD) pada Modul Lembur digunakan untuk menggambarkan struktur basis data serta hubungan antar entitas yang terlibat dalam pengelolaan lembur pegawai. ERD ini berfungsi sebagai acuan untuk memahami keterkaitan data antar tabel, sehingga alur pengolahan data lembur dapat berjalan secara konsisten dan terintegrasi. Melalui ERD, relasi antar entitas seperti pegawai, satuan kerja, periode lembur, hingga rekapitulasi nominal lembur dapat dipetakan secara jelas. Gambar 3.20 menunjukkan ERD Modul Lembur yang terdiri atas delapan tabel utama yang saling terhubung. Tabel users dan pegawai merepresentasikan identitas pengguna sistem serta data pegawai yang terlibat dalam proses lembur. Relasi antara users dan pegawai bersifat satu ke satu (one-to-one), di mana satu akun pengguna merepresentasikan satu data pegawai dalam sistem. Selanjutnya, setiap pegawai terhubung dengan satuan kerja melalui tabel satker dengan relasi satu ke banyak (one-to-many), yang menunjukkan bahwa satu satuan kerja dapat memiliki banyak pegawai, sedangkan setiap pegawai hanya terdaftar pada satu satuan kerja. Penentuan rentang waktu lembur direpresentasikan melalui tabel periode\_lembur. Relasi antara periode\_lembur dan daftar\_lembur bersifat satu ke banyak (one-to-many), di mana satu periode lembur dapat mencakup banyak data realisasi lembur pegawai. Pencatatan realisasi lembur pegawai dilakukan pada tabel daftar\_lembur, yang menyimpan informasi tanggal lembur, jam datang, jam pulang, serta jumlah jam lembur yang diajukan. Relasi antara pegawai dan daftar\_lembur juga bersifat satu ke banyak, karena satu pegawai dapat memiliki banyak catatan lembur pada periode yang berbeda. Hubungan

antara pegawai dan periode\_lembur secara konseptual membentuk relasi banyak ke banyak (many-to-many), yang direalisasikan melalui tabel daftar\_lembur sebagai tabel penghubung. Dengan demikian, satu pegawai dapat tercatat pada beberapa periode lembur, dan satu periode lembur dapat mencakup banyak pegawai. Hasil lembur yang telah divalidasi selanjutnya direkapitulasi melalui tabel rekap\_daftar\_lembur. Relasi antara periode\_lembur dan rekap\_daftar\_lembur bersifat satu ke banyak, sedangkan relasi antara pegawai dan rekap\_daftar\_lembur juga bersifat satu ke banyak, karena setiap pegawai dapat memiliki hasil rekap lembur pada lebih dari satu periode. Tabel ini memuat ringkasan lembur per pegawai dan per periode, termasuk total menit lembur serta perhitungan nominal bruto, potongan, dan nominal bersih. Selain itu, tabel dafnom\_lembur digunakan untuk menyimpan informasi dokumen nominal lembur yang dihasilkan dari proses rekapitulasi. Relasi antara rekap\_daftar\_lembur dan dafnom\_lembur bersifat satu ke satu (one-to-one), yang menunjukkan bahwa satu hasil rekap lembur menghasilkan satu dokumen nominal lembur sebagai keluaran administrasi. Dengan adanya ERD seperti yang ditunjukkan pada Gambar 3.20, relasi antar entitas pada Modul Lembur dapat dipahami dan menjadi dasar perancangan basis data agar setiap proses pengelolaan lembur berjalan terstruktur, terkontrol, dan sesuai dengan kewenangan masing-masing peran.



Gambar 3.20. Entity Relationship Diagram Modul Lembur

### C Struktur Basis Data Modul Lembur

Struktur basis data pada Modul Lembur dirancang untuk mendukung seluruh rangkaian proses pengelolaan lembur pegawai, khususnya pada tahap pemeriksaan dan validasi oleh Kepala Bagian. Basis data ini mencakup data pengguna, data pegawai, pengelompokan satuan kerja, pengaturan periode lembur, pencatatan realisasi jam lembur, hingga rekapitulasi dan pembentukan dokumen nominal lembur. Setiap tabel memiliki peran yang saling terhubung sehingga proses pengelolaan lembur dapat berjalan secara terstruktur, terkontrol, dan sesuai kewenangan.

#### 1. Tabel users

Tabel 3.7 berfungsi untuk menyimpan data akun pengguna aplikasi yang digunakan dalam proses autentikasi dan pengendalian akses. Melalui tabel ini, aplikasi dapat mengenali identitas pengguna yang masuk ke dalam sistem beserta perannya, termasuk peran Kepala Bagian. Kolom email digunakan sebagai penghubung antara akun pengguna dan data pegawai, sehingga akses terhadap data lembur dapat dibatasi sesuai struktur organisasi dan unit kerja masing-masing.

Tabel 3.7. Struktur tabel users

Atribut	Tipe Data	Keterangan
id	INT	ID user, primary key, auto increment.
nama	VARCHAR	Nama pengguna akun.
email	VARCHAR	Email akun, penghubung ke tabel pegawai.
idbiro	INT	Identitas biro pengguna (jika digunakan).
idbagian	INT	Identitas bagian pengguna.
role	VARCHAR	Peran pengguna dalam sistem.

## 2. Tabel pegawai

Tabel 3.8 merupakan tabel induk yang menyimpan data seluruh pegawai. Data ini menjadi dasar dalam proses pemeriksaan lembur oleh Kepala Bagian, karena setiap pengajuan lembur selalu dikaitkan dengan data pegawai tertentu. Melalui kolom id\_satker, setiap pegawai terhubung dengan satu satuan kerja sehingga pemfilteran data lembur dapat dilakukan berdasarkan unit kerja yang berada di bawah kewenangan Kepala Bagian.

Tabel 3.8. Struktur tabel pegawai

Atribut	Tipe Data	Keterangan
id	INT	ID pegawai, primary key.
nip	VARCHAR	Nomor Induk Pegawai.
nama	VARCHAR	Nama pegawai.
email	VARCHAR	Email pegawai, terhubung dengan akun user.
id_satker	INT	Relasi ke tabel satker.

## 3. Tabel satker

Tabel 3.9 menyimpan data satuan kerja resmi dalam organisasi. Data pada tabel ini digunakan sebagai acuan pembatasan akses data lembur, sehingga Kepala Bagian hanya dapat melihat dan memeriksa lembur pegawai yang berada pada satuan kerja yang dipimpinnya. Satu satuan kerja dapat memiliki banyak pegawai dan data lembur yang terkait.

Tabel 3.9. Struktur tabel satker

Atribut	Tipe Data	Keterangan
id_satker	INT	ID satuan kerja, primary key.
uraian	VARCHAR	Nama atau uraian satuan kerja.

4. Tabel periode\_lembur

Tabel 3.10 digunakan untuk menyimpan informasi rentang waktu lembur yang berlaku. Data periode ini menjadi acuan bagi Kepala Bagian dalam memilih periode tertentu saat melakukan pemeriksaan dan koreksi data lembur pegawai. Dengan adanya tabel ini, pengelolaan lembur dapat dilakukan secara terkelompok berdasarkan periode yang jelas.

Tabel 3.10. Struktur tabel periode\_lembur

Atribut	Tipe Data	Keterangan
id	INT	ID periode, primary key.
tanggal_awal	DATE	Tanggal awal periode lembur.
tanggal_akhir	DATE	Tanggal akhir periode lembur.
status	VARCHAR	Status periode lembur.

5. Tabel jadwal\_lembur

Tabel 3.11 menyimpan rencana pelaksanaan lembur pegawai sebelum realisasi dilakukan. Data ini berfungsi sebagai referensi awal dalam proses pemeriksaan, sehingga Kepala Bagian dapat membandingkan antara rencana lembur dan realisasi jam lembur yang dicatat oleh pegawai.

Tabel 3.11. Struktur tabel jadwal\_lembur

Atribut	Tipe Data	Keterangan
id	INT	ID jadwal lembur, primary key.
id_pegawai	INT	Relasi ke tabel pegawai.
tanggal	DATE	Tanggal rencana lembur.
status	VARCHAR	Status jadwal lembur.

6. Tabel daftar\_lembur

Tabel 3.12 merupakan tabel utama dalam proses pemeriksaan lembur oleh Kepala Bagian. Tabel ini menyimpan data realisasi lembur pegawai, termasuk jam datang, jam pulang, serta total jam lembur. Melalui tabel ini, Kepala Bagian melakukan pemeriksaan kebenaran jam lembur dan memberikan koreksi apabila ditemukan ketidaksesuaian, yang ditandai melalui kolom isCheck dan isEdit.

Tabel 3.12. Struktur tabel daftar\_lembur

Atribut	Tipe Data	Keterangan
id	INT	ID daftar lembur, primary key.
id_pegawai	INT	Pegawai yang melakukan lembur.
id_satker	INT	Satuan kerja pegawai.
id_periode	INT	Periode lembur.
tanggal	DATE	Tanggal realisasi lembur.
jam_datang	TIME	Jam mulai lembur.
jam_pulang	TIME	Jam selesai lembur.
jam_lembur	INT	Total jam lembur.
isCheck	TINYINT	Penanda data telah diperiksa.
isEdit	TINYINT	Penanda data mengalami koreksi.

7. Tabel rekap\_daftar\_lembur

Tabel 3.13 digunakan untuk menyimpan hasil rekapitulasi lembur per pegawai dalam satu periode tertentu. Data pada tabel ini menyajikan ringkasan lembur yang telah diperiksa, sehingga Kepala Bagian dapat melihat total akumulasi waktu lembur serta nilai nominal sebelum proses pembentukan dokumen nominal lembur dilakukan.

Tabel 3.13. Struktur tabel rekap\_daftar\_lembur

Atribut	Tipe Data	Keterangan
id	INT	ID rekap, primary key.
id_pegawai	INT	Pegawai terkait.
id_periode	INT	Periode lembur.
total_menit	INT	Akumulasi menit lembur.
nominal_bruto	DECIMAL	Total nominal lembur.
potongan	DECIMAL	Nilai potongan.
nominal_bersih	DECIMAL	Nominal akhir lembur.

8. Tabel dafnom\_lembur

Tabel 3.14 menyimpan informasi dokumen nominal lembur yang dihasilkan setelah proses pemeriksaan dan rekapitulasi selesai dilakukan. Data pada tabel ini merepresentasikan tahap akhir pengelolaan lembur, yang selanjutnya digunakan sebagai dasar proses administrasi lanjutan dan penandatanganan dokumen.



Tabel 3.14. Struktur tabel dafnom\_lembur

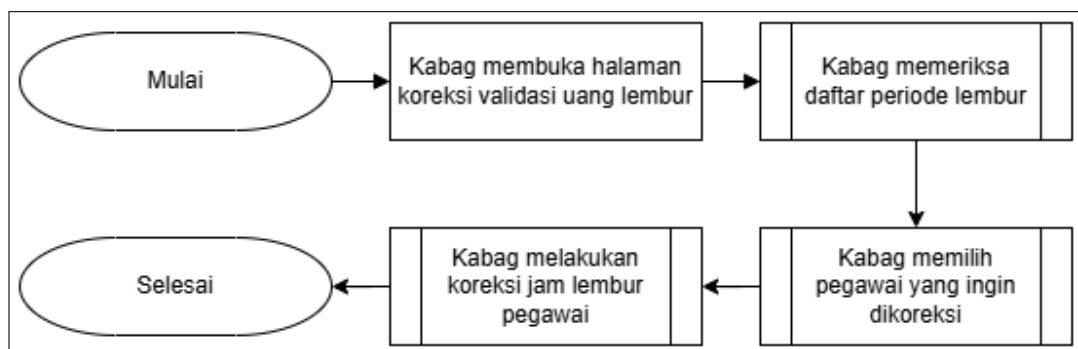
Atribut	Tipe Data	Keterangan
id	INT	ID dokumen nominal, primary key.
id_satker	INT	Satuan kerja terkait.
tanggal_awal	DATE	Awal periode dokumen.
tanggal_akhir	DATE	Akhir periode dokumen.
no_lembur	VARCHAR	Nomor dokumen lembur.
path	VARCHAR	Lokasi penyimpanan dokumen.
updated_at	DATETIME	Waktu pembaruan terakhir.

#### D Alur Proses Modul Lembur (Peran Kabag)

Proses kerja dalam Modul Lembur untuk peran *Kabag* mencakup empat bagian utama, yaitu alur umum koreksi lembur, pemeriksaan daftar periode lembur, pemilihan pegawai yang ingin dikoreksi, serta proses koreksi jam lembur pegawai. Setiap bagian dijelaskan melalui flowchart berikut.

##### 1. Alur umum proses Modul Lembur Kabag

Gambar 3.21 memperlihatkan gambaran besar proses kerja Kabag dalam modul lembur. Proses dimulai ketika Kabag membuka halaman koreksi dan validasi uang lembur. Kabag kemudian memeriksa daftar periode lembur yang telah tercatat di sistem, memilih pegawai yang ingin dikoreksi, dan melakukan penyesuaian terhadap jam lembur pegawai sesuai kebutuhan. Setelah proses koreksi selesai, Kabag dapat menyimpan hasil koreksi tersebut sebagai data valid akhir.

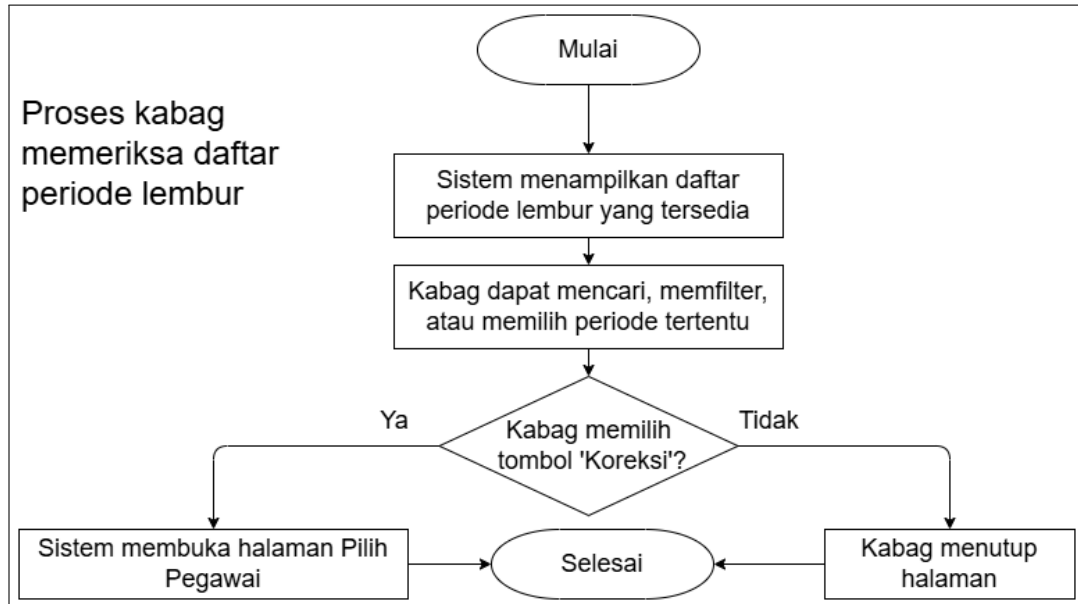


Gambar 3.21. Flowchart alur umum Modul Lembur Kabag

##### 2. Proses pemeriksaan daftar periode lembur

Gambar 3.22 menggambarkan tahapan saat Kabag memeriksa daftar periode lembur yang tersedia. Sistem akan menampilkan seluruh periode lembur aktif pada halaman utama. Kabag dapat melakukan pencarian, penyaringan, maupun memilih periode tertentu untuk diperiksa lebih lanjut. Jika Kabag menekan tombol “Koreksi”, sistem

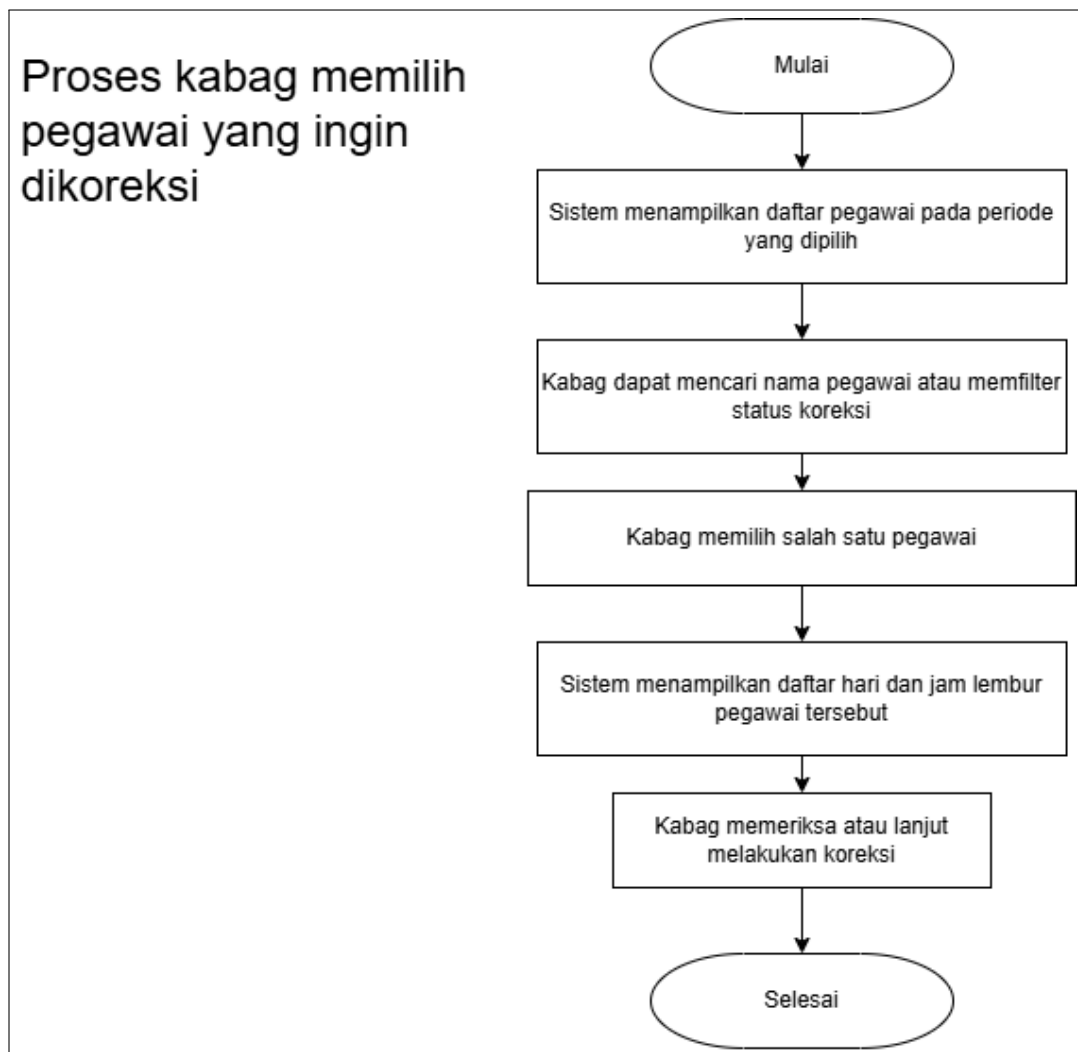
akan membuka halaman pemilihan pegawai. Namun, jika tidak ada periode yang ingin dikoreksi, Kabag dapat menutup halaman dan proses berakhir pada tahap ini.



Gambar 3.22. Flowchart proses pemeriksaan daftar periode lembur

### 3. Proses pemilihan pegawai yang ingin dikoreksi

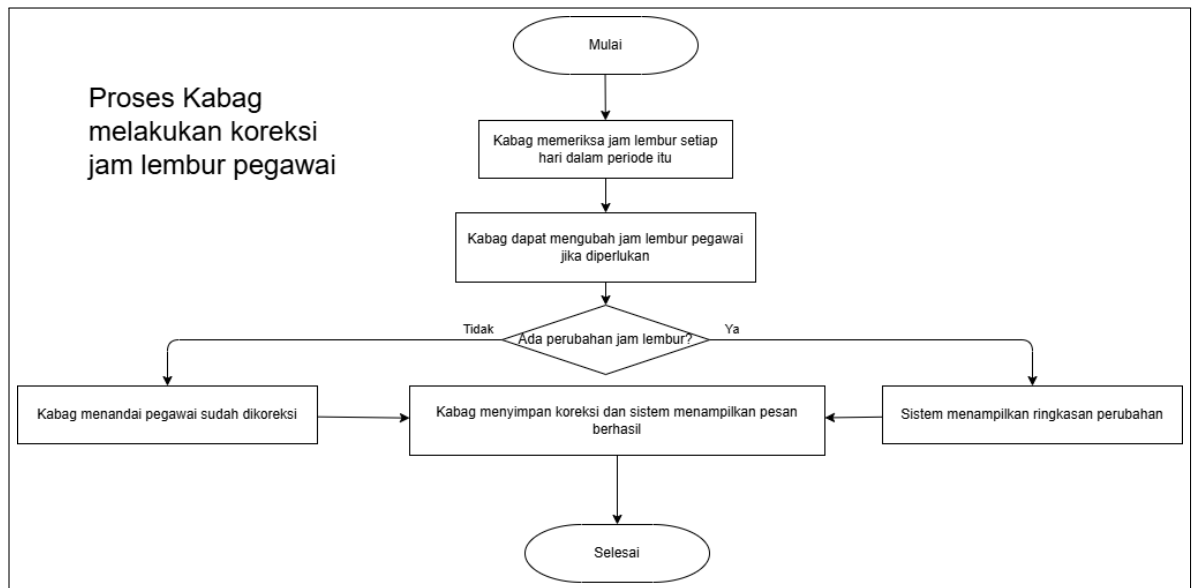
Gambar 3.23 menunjukkan tahapan di mana Kabag menentukan pegawai mana yang perlu dilakukan koreksi. Setelah memilih periode lembur, sistem akan menampilkan daftar pegawai yang termasuk dalam periode tersebut. Kabag dapat mencari nama pegawai atau memfilter berdasarkan status koreksi untuk mempermudah pencarian. Setelah pegawai dipilih, sistem menampilkan daftar hari dan jam lembur pegawai tersebut, dan Kabag dapat memeriksa atau melanjutkan ke tahap koreksi.



Gambar 3.23. Flowchart proses pemilihan pegawai yang ingin dikoreksi

#### 4. Proses Kabag melakukan koreksi jam lembur pegawai

Gambar 3.24 memperlihatkan alur detail ketika Kabag melakukan koreksi terhadap jam lembur pegawai. Kabag memeriksa jam lembur pegawai per hari dalam periode yang dipilih. Bila diperlukan, Kabag dapat mengubah jam lembur tersebut sesuai hasil verifikasi. Jika tidak ada perubahan, Kabag dapat langsung menandai bahwa pegawai tersebut sudah dikoreksi. Namun, jika terdapat perubahan, sistem akan menampilkan ringkasan perubahan sebelum data disimpan. Setelah Kabag menyimpan koreksi, sistem menampilkan pesan bahwa proses koreksi telah berhasil dan data tersimpan secara permanen.



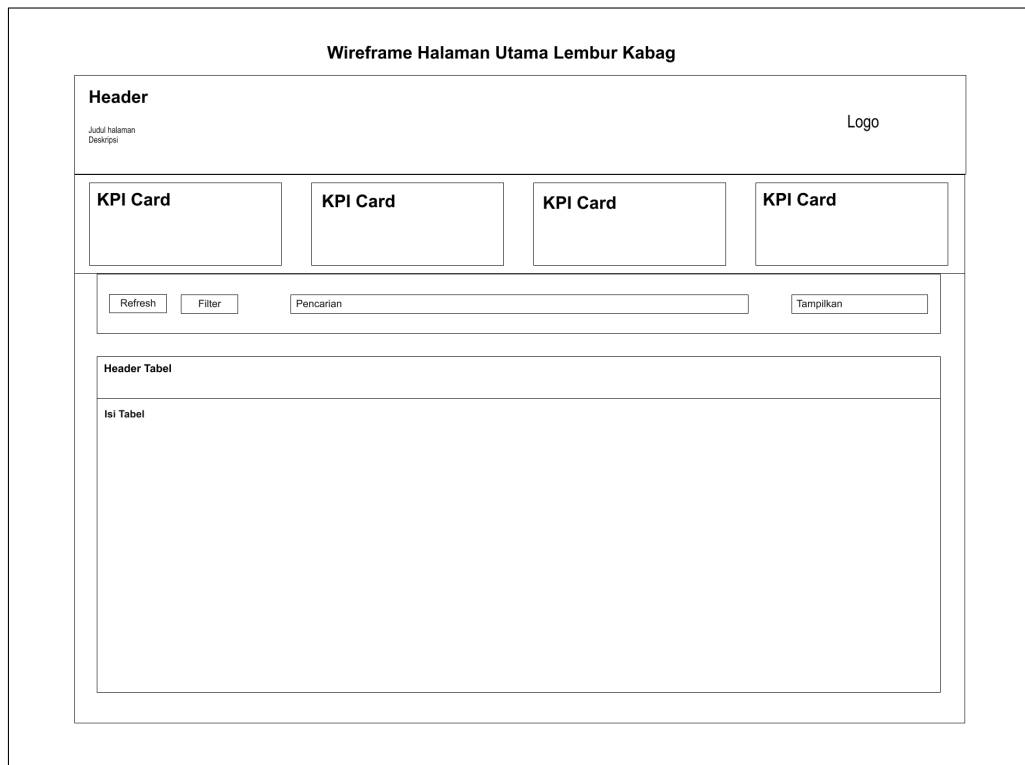
Gambar 3.24. Flowchart proses Kabag melakukan koreksi jam lembur pegawai

## E Wireframe Modul Lembur (Peran Kabag)

*Wireframe* digunakan untuk memetakan tata letak antarmuka sebelum tahap implementasi dilakukan. Pada Modul Lembur untuk peran Kepala Bagian (Kabag), *wireframe* berfungsi sebagai acuan penyusunan komponen halaman yang mendukung proses peninjauan data lembur, pemilihan pegawai, hingga pemeriksaan rincian lembur per pegawai. Dengan adanya *wireframe* ini, implementasi antarmuka pada tahap frontend dapat dilakukan secara konsisten sesuai kebutuhan alur kerja Kabag.

### 1. Halaman Utama Lembur Kabag

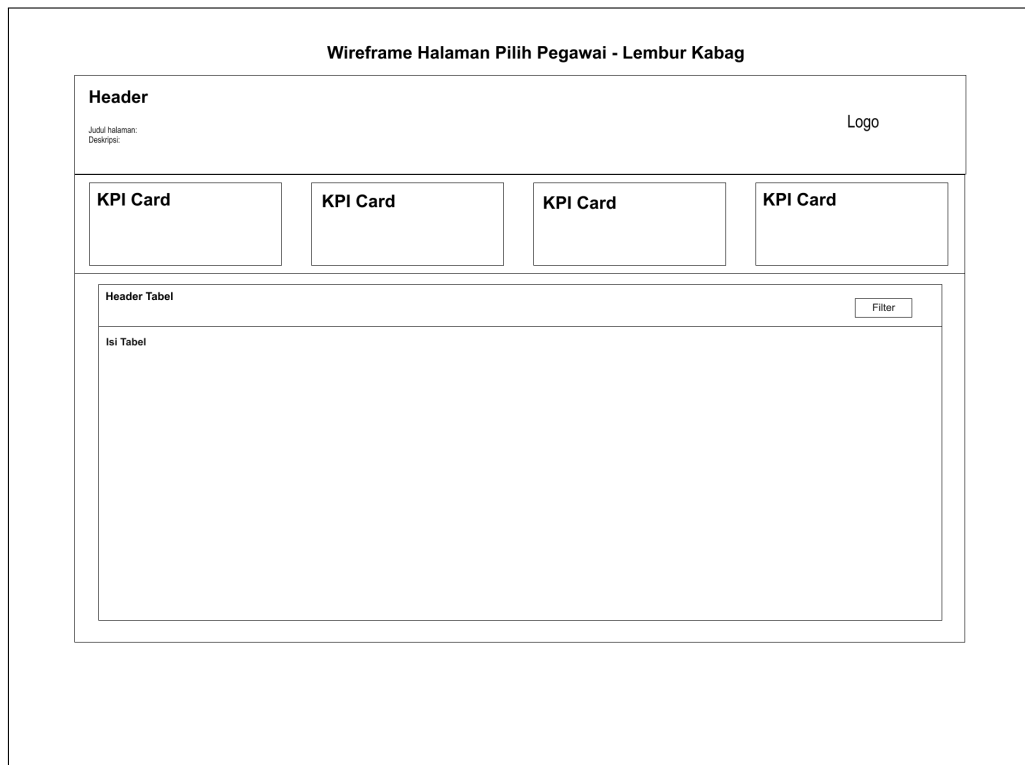
Gambar 3.25 menampilkan *wireframe* halaman utama Lembur Kabag yang digunakan sebagai acuan tampilan awal saat Kabag mengakses modul. *Wireframe* ini menggambarkan susunan komponen yang membantu Kabag memantau ringkasan kondisi data lembur dan melihat daftar periode lembur yang tersedia. Secara umum, halaman ini terdiri dari header halaman yang memuat judul dan deskripsi singkat, bagian ringkasan dalam bentuk kartu KPI, bilah aksi yang memuat tombol Refresh dan Filter, kolom pencarian, serta kontrol Tampilan. Di bawahnya terdapat tabel utama yang menampilkan daftar periode lembur beserta informasi status dan aksi untuk melanjutkan ke proses pemeriksaan data.



Gambar 3.25. *Wireframe* halaman utama Lembur Kabag

## 2. Halaman Pilih Pegawai

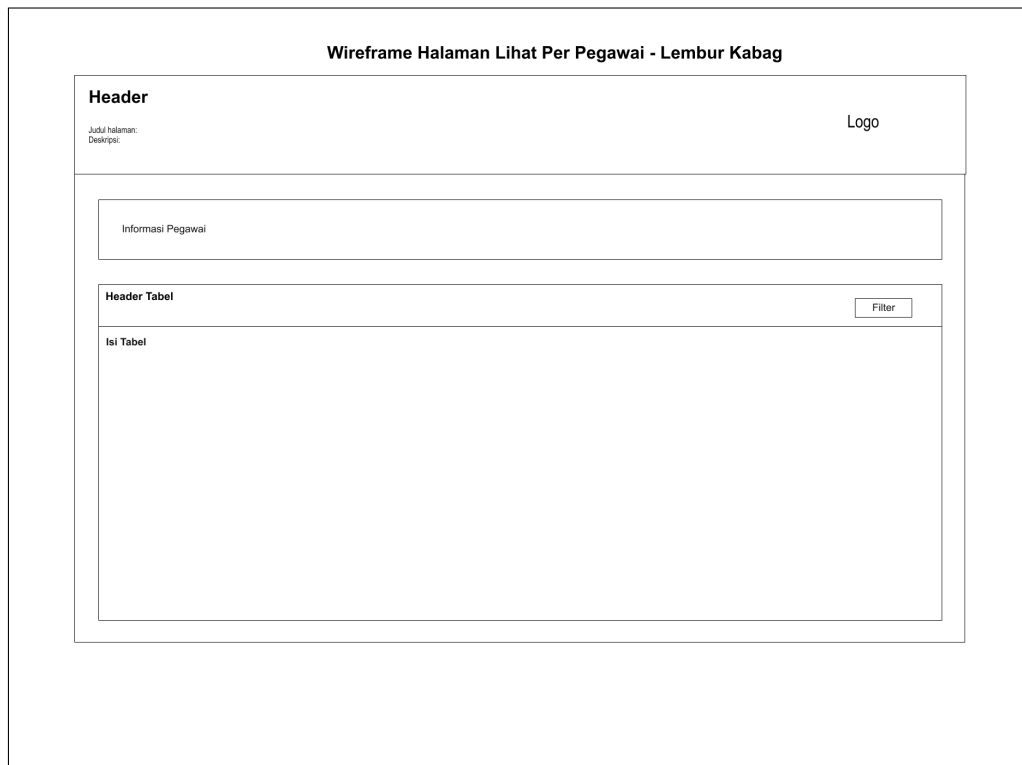
Gambar 3.26 menampilkan *wireframe* halaman Pilih Pegawai yang digunakan setelah Kabag memilih salah satu periode lembur. *Wireframe* ini menggambarkan halaman yang berfungsi untuk menampilkan daftar pegawai pada periode terpilih agar Kabag dapat menentukan pegawai yang akan diperiksa. Komponen utama pada halaman ini meliputi header halaman, ringkasan informasi dalam bentuk kartu KPI, serta tabel daftar pegawai. Pada bagian tabel, tersedia tombol Filter untuk membantu penyaringan data sehingga Kabag dapat menemukan pegawai yang relevan dengan lebih cepat.



Gambar 3.26. *Wireframe* halaman pilih pegawai pada Lembur Kabag

### 3. Halaman Lihat Per Pegawai

Gambar 3.27 menampilkan *wireframe* halaman Lihat Per Pegawai yang digunakan untuk meninjau rincian data lembur harian milik seorang pegawai dalam periode tertentu. *Wireframe* ini menggambarkan susunan komponen yang mendukung proses pemeriksaan data secara rinci, dimulai dari bagian Informasi Pegawai yang menampilkan identitas pegawai, hingga tabel utama yang berisi daftar lembur harian. Pada bagian header tabel tersedia tombol Filter untuk membantu Kabag melakukan penyaringan data pada tampilan rincian tersebut.



Gambar 3.27. *Wireframe* halaman lihat per pegawai pada Lembur Kabag

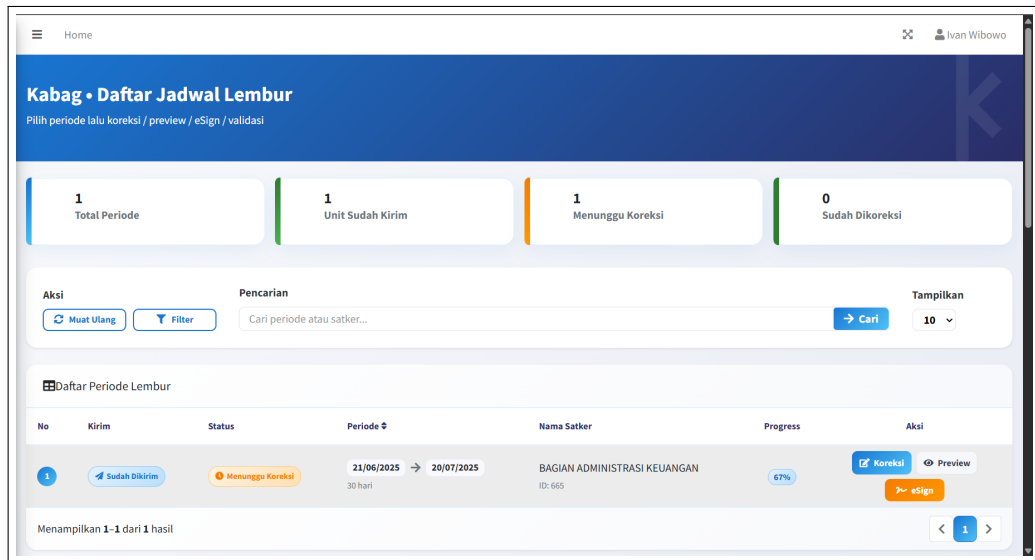
Ketiga *wireframe* di atas menjadi acuan dasar dalam pengembangan tampilan frontend Modul Lembur untuk peran Kabag, sehingga alur peninjauan periode, pemilihan pegawai, dan pemeriksaan rincian lembur dapat ditampilkan secara terstruktur dan mudah dipahami.

## F Implementasi Antarmuka (Frontend)

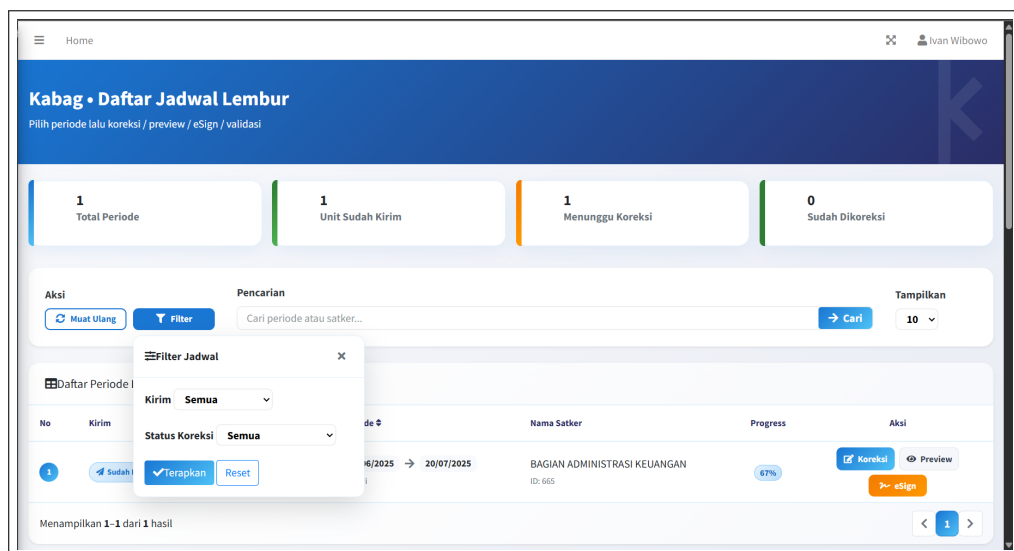
Antarmuka pengguna *frontend* pada Modul Lembur (Peran Kabag) dirancang dengan pendekatan sederhana, terstruktur, dan informatif agar setiap tahapan koreksi lembur dapat dilakukan secara mudah. Seluruh halaman dibangun menggunakan *Blade Template* Laravel dengan dukungan *DataTables*, *Font Awesome Icons*, dan gaya visual khas DIGITall Keuangan yakni gradasi biru lembut (#186FC4 hingga #A9D3F5) dengan elemen putih bersih untuk menjaga keterbacaan. Bagian antarmuka terdiri atas beberapa komponen utama berikut.

1. Halaman utama: header, ringkasan statistik, dan tabel daftar periode lembur  
Pada bagian atas halaman terlihat pada Gambar 3.28, sistem menampilkan judul “Kabag • Daftar Jadwal Lembur” beserta deskripsi singkat untuk memandu pengguna. Di bawahnya terdapat empat kartu informasi yang menampilkan jumlah total periode lembur, unit yang telah mengirim data, status menunggu koreksi, serta data yang telah dikoreksi. Bagian utama halaman menampilkan tabel daftar periode lembur dengan kolom “Kirim”, “Status”, “Periode”, “Nama Satker”, “Progress”,

dan “Aksi”. Setiap baris tabel menampilkan status pengiriman (misalnya “Sudah Dikirim”) dan status koreksi (seperti “Menunggu Koreksi” atau “Sudah Dikoreksi”), yang ditandai dengan badge berwarna dinamis agar mudah diidentifikasi. Pengguna dapat menggunakan kotak pencarian untuk menemukan periode tertentu atau satker terkait, serta menekan tombol “Filter” untuk menyaring berdasarkan status kirim dan koreksi seperti pada Gambar 3.29.



Gambar 3.28. Halaman utama Modul Lembur Kabag - daftar periode lembur



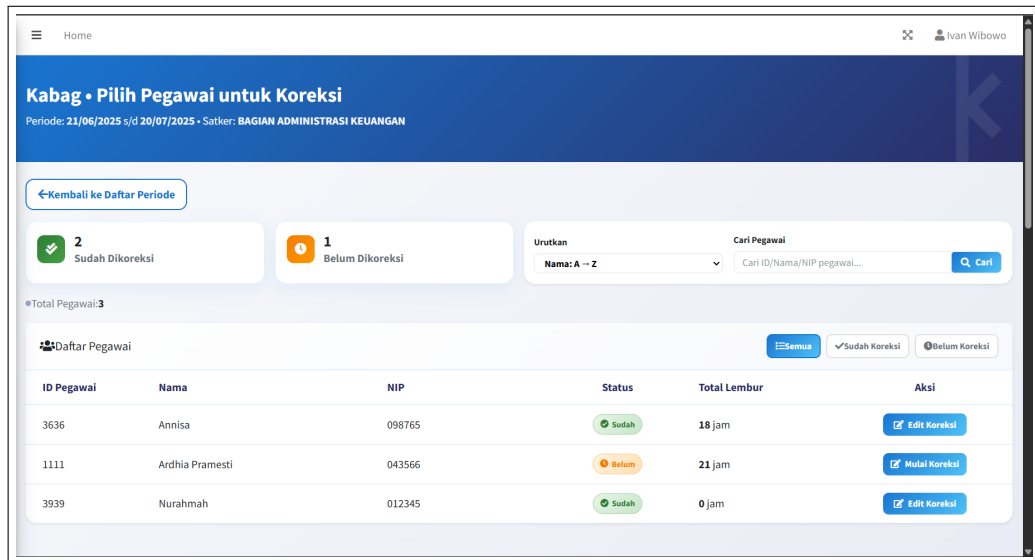
Gambar 3.29. Fitur filter periode lembur berdasarkan status kirim dan koreksi

## 2. Halaman daftar pegawai per periode

Setelah Kabag memilih periode tertentu dan menekan tombol “Koreksi”, sistem



menampilkan halaman daftar pegawai pada periode tersebut seperti terlihat pada Gambar 3.30. Tabel ini memuat ID pegawai, nama, NIP, status koreksi (Sudah/Belum), serta total jam lembur yang tercatat. Warna hijau digunakan untuk status “Sudah”, sedangkan oranye menandai “Belum”.



Gambar 3.30. Halaman daftar pegawai pada periode lembur yang dipilih

### 3. Halaman koreksi jam lembur

Setelah memilih salah satu pegawai, Kabag diarahkan ke halaman “Koreksi Jam Lembur” dapat dilihat pada Gambar 3.31. Pada bagian atas, sistem menampilkan identitas pegawai, NIP, serta periode lembur yang sedang dikoreksi. Di bawahnya terdapat tabel jam lembur harian dengan kolom tanggal, jam saat ini, jam koreksi, status lembur, dan status perubahan. Nilai jam koreksi dapat diubah langsung dengan input angka. Perubahan yang dilakukan akan ditandai dengan warna kuning dan label “Diubah” agar Kabag mudah mengenali data yang telah dimodifikasi.

Employee: Ardhia Pramesti  
ID: 1111 NIP: 043566 21/06/2025 - 20/07/2025  
Total Jam (tampil): 20  
Baris Diubah: 1

Edt nilai jam langsung pada kolom "Jam Koreksi" (tidak boleh melebihi jam saat ini)

TANGGAL	JAM SAAT INI	JAM KOREKSI	STATUS LEMBUR	STATUS
21/06/2025 SAB	1 jam	1 jam Max: 1	Lembur	Original
22/06/2025 MIN	0 jam	0 jam Max: 0	Normal	Original
23/06/2025 SEN	0 jam	0 jam Max: 0	Normal	Original
24/06/2025 SEL	2 jam	1 jam Max: 2	Lembur	Diubah
25/06/2025 RAB	0 jam	0 jam Max: 0	Normal	Original
26/06/2025	0 jam	0 jam	Normal	Original

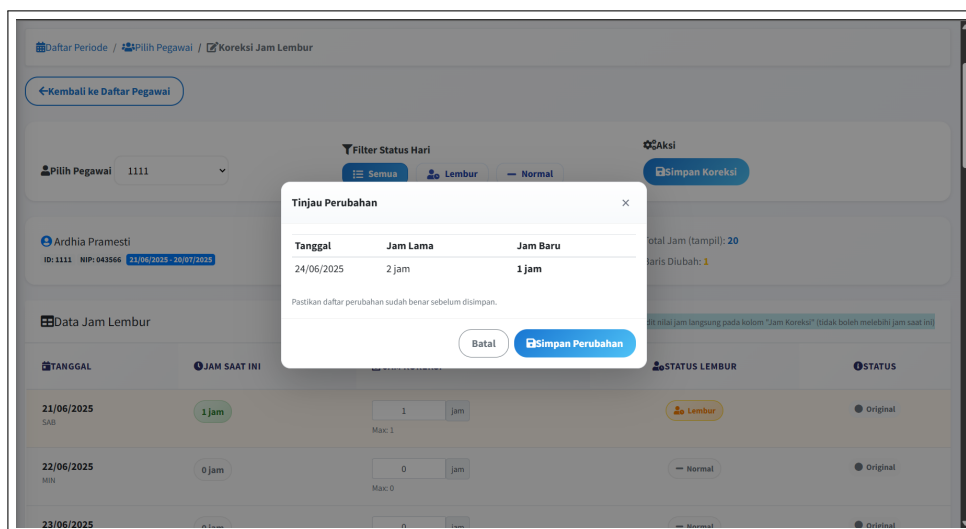
Gambar 3.31. Tabel jam lembur harian dengan kolom koreksi dan status perubahan

#### 4. Validasi koreksi: dialog konfirmasi, notifikasi, dan pembaruan indikator

Untuk memastikan koreksi dilakukan dengan benar, sistem menyediakan rangkaian validasi dan umpan balik berikut.

- Dialog konfirmasi koreksi

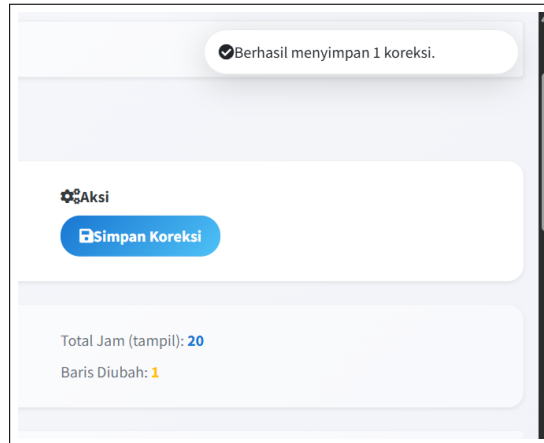
Sebelum penyimpanan, sistem menampilkan jendela konfirmasi berisi daftar perubahan yang dilakukan, seperti tanggal lembur, jam lama, dan jam baru seperti yang terlihat pada Gambar 3.32. Kabag dapat meninjau ulang data sebelum menekan tombol “Simpan Perubahan”. Proses ini menambah lapisan keamanan agar koreksi yang dilakukan benar-benar valid.



Gambar 3.32. Dialog konfirmasi koreksi jam lembur sebelum penyimpanan

- Validasi dan notifikasi keberhasilan

Setelah data disimpan, sistem menampilkan notifikasi *toast* bertuliskan “Berhasil menyimpan 1 koreksi” di bagian atas halaman seperti pada Gambar 3.33. Indikator “Baris Diubah” serta total jam lembur juga diperbarui secara otomatis di bawah tabel sebagai umpan balik langsung kepada pengguna.



Gambar 3.33. Notifikasi keberhasilan dan indikator baris diubah setelah koreksi

## G Implementasi *Backend*

Bagian ini memerinci logika pada Modul Lembur (peran Kabag) dalam bentuk pseudocode.

### 1. Pemrosesan Token Periode

Proses ini berfungsi untuk membaca dan menerjemahkan token periode yang dikirim dari portal utama DPR RI ke sistem DIGITall. Token tersebut berisi informasi rentang tanggal serta identitas satuan kerja (id satker) yang telah dienkripsi dalam format base64 agar aman saat dikirim melalui URL. Sebelum sistem menampilkan data lembur, token ini harus diproses terlebih dahulu untuk memastikan nilai yang diterima valid dan tidak mengalami perubahan selama transmisi. Fungsi ini bekerja dengan cara mengubah kembali token terenkripsi menjadi tiga nilai utama, yaitu tanggal awal, tanggal akhir, dan id satker. Setelah itu dilakukan validasi struktur dan format tanggal agar sistem hanya memproses data yang sesuai dengan periode yang sah. Apabila token rusak atau tidak memenuhi format, sistem akan menolak permintaan dengan menampilkan galat 404. Langkah kerja fungsi tersebut dijelaskan pada Pseudocode 3.6.

```
1 FUNCTION bacaTokenPeriode(token):  
2     // 1. Hilangkan karakter URL encoding dari token  
3     teksAsli = urlDecode(token)  
4  
5     // 2. Coba ubah token base64 ke bentuk teks biasa  
6     hasil = base64Decode(teksAsli)  
7     IF hasil GAGAL:
```

```

8      // Ubah karakter varian URL-safe (-_ -> +/) dan
      tambah padding (=)
9      teksNormal = gantiKarakter(teksAsli, "-",
      "+/")
10     hasil = base64Decode(tambahPadding(teksNormal))
11     ENDIF
12
13     // 3. Jika masih gagal, tolak akses
14     IF hasil GAGAL:
15         ERROR 404 "Token periode tidak valid"
16
17     // 4. Pisahkan hasil menjadi tiga bagian: awal|
      akhir|id_satker
18     bagian = pisahString(hasil, "|")
19     IF jumlah(bagian)      3:
20         ERROR 404 "Format token salah"
21
22     // 5. Konversi nilai tanggal ke format standar ISO
23     tanggalAwal = formatTanggalISO(parseTanggal(bagian
      [0]))
24     tanggalAkhir = formatTanggalISO(parseTanggal(bagian
      [1]))
25     idSatker      = bagian[2]
26
27     RETURN [tanggalAwal, tanggalAkhir, idSatker]
28 END FUNCTION

```

Kode 3.6: Pseudocode pembacaan token periode lembur

Fungsi `bacaTokenPeriode` memastikan sistem hanya menampilkan data lembur yang sesuai dengan periode resmi yang dikirim dari portal DPR RI. Dengan demikian, setiap halaman yang menampilkan data lembur akan selalu menggunakan parameter yang terverifikasi dan konsisten.

Sebagai contoh penggunaan, ketika pengguna membuka halaman detail lembur, sistem akan menerima token periode dari URL. Token tersebut kemudian diproses menggunakan fungsi ini untuk mendapatkan rentang tanggal dan identitas satuan kerja. Hasilnya dipakai oleh modul Lembur untuk menampilkan daftar data sesuai periode yang dimaksud.

## 2. Daftar Periode dan Ringkasan KPI

Fungsi ini digunakan untuk menampilkan daftar periode lembur yang dimiliki oleh satu satuan kerja beserta ringkasan indikator kinerja utama (KPI) pada setiap periode. Data yang diambil meliputi status pengiriman, tingkat progres koreksi, dan status validasi. Informasi ini menjadi dasar bagi Kepala Bagian (Kabag) untuk memantau sejauh mana proses koreksi lembur telah dilakukan pada tiap periode.

Prosesnya diawali dengan pengambilan seluruh data periode lembur dari tabel periode untuk satu satker tertentu. Setiap periode kemudian dihitung jumlah pegawai yang tercatat serta jumlah pegawai yang sudah menyelesaikan proses koreksi. Dari hasil perhitungan tersebut sistem menentukan nilai progres dalam bentuk persentase dan menandai statusnya sebagai “Pending” jika belum penuh atau “Validated” jika seluruh pegawai telah dikoreksi. Langkah kerja lengkapnya digambarkan pada Pseudocode 3.7.

```

1 FUNCTION daftarPeriode(idSatker):
2     // 1. Ambil seluruh periode lembur milik satker
3     periodeList = ambilSemua("periode_lembur", idSatker
4     )
5
6     hasil = []
7     UNTUK setiap periode DALAM periodeList:
8         // 2. Hitung total pegawai dan jumlah yang
9         sudah dikoreksi
10        totalPegawai = hitungUnik("daftar_lembur",
11        idSatker, periode)
12        sudahKoreksi = hitungUnik("daftar_lembur",
13        idSatker, periode, isCheck=1)
14
15        // 3. Hitung persentase progres
16        progres = persen(sudahKoreksi, totalPegawai)
17
18        // 4. Tentukan status validasi
19        status = JIKA progres >= 100 MAKA "Validated"
20        LAINNYA "Pending"
21
22        // 5. Simpan hasil ke daftar keluaran
23        hasil += {
24            idToken : encodeBase64(periode.awal + "|" +
25            periode.akhir + "|" + idSatker),
26            rentang : [periode.awal, periode.akhir],
27            kirim    : (periode.status==1 ? "Sudah Dikirim
28            " : "Belum Dikirim"),
29            progres  : progres,
30            status   : status
31        }
32
33        KEMBALIKAN hasil
34    END UNTUK
35 END FUNCTION

```

Kode 3.7: Pseudocode daftar periode dan ringkasan KPI

Fungsi ini membantu Kabag melihat perkembangan koreksi lembur secara menyeluruh dalam satu tampilan. Nilai progres dan status validasi ditampilkan pada tabel utama dan kartu KPI agar mudah dipantau.

Sebagai contoh penggunaan, ketika halaman Kabag pertama kali dimuat, sistem akan memanggil fungsi ini melalui *endpoint* daftar periode. Hasilnya digunakan untuk mengisi tabel daftar periode dan menampilkan ringkasan KPI pada bagian atas halaman.

### 3. Daftar Pegawai per Periode

Fungsi ini digunakan untuk menampilkan daftar seluruh pegawai yang terlibat dalam satu periode lembur. Data yang disajikan meliputi nama, NIP, total jam lembur, status koreksi, serta jumlah baris data yang telah diubah. Informasi ini membantu Kabag mengetahui siapa saja pegawai yang sudah dikoreksi dan siapa yang masih perlu ditinjau ulang. Prosesnya dilakukan dengan mengambil seluruh data lembur pada rentang tanggal tertentu untuk satu satker. Sistem kemudian menentukan satu baris referensi per pegawai agar identitas seperti nama dan NIP selalu konsisten meskipun ada perubahan pada data hari lain. Setelah itu dilakukan perhitungan jumlah jam lembur, status koreksi, serta jumlah baris yang diubah. Urutan hasil disusun berdasarkan ID pegawai untuk menjaga stabilitas tampilan. Langkahnya digambarkan pada Pseudocode 3.8.

```
1 FUNCTION daftarPegawai(awal, akhir, idSatker):
2     // 1. Ambil baris referensi per pegawai
3     refMin = ambilTanggalTerkecilPerPegawai(awal, akhir
4         , idSatker)
5
6     // 2. Hitung total jam, status koreksi, dan jumlah
7     baris diubah
8     hasil = []
9     UNTUK setiap pegawai DALAM refMin:
10         totalJam      = jumlahJamLembur(pegawai, awal,
11             akhir, idSatker)
12         sudahKoreksi   = cekApakahSudahKoreksi(pegawai,
13             awal, akhir, idSatker)
14         totalDiubah    = hitungBarisEdit(pegawai, awal,
15             akhir, idSatker)
16
17         hasil += {
18             idPegawai      : pegawai.id,
19             nama            : pegawai.nama,
20             nip             : pegawai.nip,
21             totalJam        : totalJam,
22             sudahKoreksi    : sudahKoreksi,
23             totalBarisEdit  : totalDiubah
24         }
```

```

20     KEMBALIKAN hasil
21 END FUNCTION

```

### Kode 3.8: Pseudocode daftar pegawai dalam periode lembur

Fungsi ini memastikan identitas pegawai selalu tampil konsisten di seluruh halaman koreksi. Sebagai contoh penggunaan, setelah Kabag memilih satu periode dan menekan tombol Koreksi, sistem memanggil fungsi ini untuk mengisi tabel daftar pegawai pada halaman tersebut.

#### 4. Data Per Pegawai untuk Koreksi

Fungsi ini berperan untuk menampilkan data lembur per hari dari satu pegawai dalam periode tertentu. Data ini digunakan oleh Kabag saat melakukan proses koreksi, menampilkan tanggal lembur, jumlah jam, serta status apakah baris tersebut pernah diubah atau belum. Proses dimulai dengan validasi parameter yang dikirim, seperti ID pegawai, periode, dan satuan kerja. Setelah itu sistem mengambil seluruh baris lembur yang sesuai dari basis data dan mengonversi nilainya ke format yang seragam agar mudah ditampilkan di antarmuka. Bila data tidak ditemukan, sistem akan mengembalikan daftar kosong. Alurnya digambarkan pada Pseudocode 3.9.

```

1 FUNCTION dataLemburPegawai(idPegawai, awal, akhir,
  idSatker):
2     // 1. Validasi parameter wajib
3     pastikanAda(idPegawai, awal, akhir, idSatker)
4
5     // 2. Ambil semua data lembur dalam periode
6     data = ambilDataLembur(idPegawai, idSatker, awal,
  akhir)
7
8     // 3. Ubah format agar seragam
9     hasil = []
10    UNTUK setiap baris DALAM data:
11        hasil += {
12            tanggal    : ubahKeTanggalISO(baris.tanggal),
13            jamLembur   : toInt(baris.jam_lembur),
14            sudahEdit   : toBool(baris.is_edit),
15            nama        : baris.nama,
16            nip         : baris.nip
17        }
18
19    KEMBALIKAN hasil
20 END FUNCTION

```

### Kode 3.9: Pseudocode data lembur per pegawai

Fungsi ini memastikan setiap data lembur per pegawai tersaji secara detail dan siap untuk dikoreksi. Sebagai contoh penggunaan, ketika Kabag memilih satu

pegawai pada halaman koreksi, sistem akan memanggil fungsi ini untuk mengisi tabel lembur harian pegawai tersebut.

##### 5. Penyimpanan Koreksi dan Penandaan

Fungsi ini digunakan untuk menyimpan perubahan jam lembur yang telah dikoreksi atau sekadar menandai bahwa data pegawai sudah diperiksa tanpa mengubah nilainya. Mekanisme ini penting agar sistem dapat mencatat progres koreksi secara akurat. Prosesnya dilakukan dalam dua mode: jika Kabag hanya ingin menandai data tanpa perubahan, sistem langsung memperbarui status koreksi. Namun, bila terdapat perubahan jam, sistem memvalidasi data baru, memastikan nilainya tidak melebihi jam lembur sebelumnya, dan melakukan pembaruan secara transaksional agar tidak ada data setengah tersimpan. Setiap baris yang diperbarui juga ditandai sebagai data hasil koreksi. Langkah detailnya dijelaskan pada Pseudocode 3.10.

```
1 FUNCTION simpanKoreksi(awal, akhir, idSatker, idPegawai
  , hanyaTandai, daftarPerubahan):
2   pastikanPegawaiAda(awal, akhir, idSatker, idPegawai
  )
3
4   // Mode 1: hanya tandai sudah dikoreksi
5   JIKA hanyaTandai == true:
6       tandaiSemuaSebagaiDiperiksa(idSatker, idPegawai
  , awal, akhir)
7       KEMBALIKAN { ok:true, pesan:"Data sudah
  ditandai dikoreksi" }
8
9   // Mode 2: simpan perubahan jam
10  mulaiTransaksi()
11  hasil = { diubah:0, sama:0, tidakDitemukan:0,
  pelanggaran:[] }
12
13  UNTUK setiap entri DALAM daftarPerubahan:
14      jamSekarang = ambilJamLembur(entri.tanggal,
  idPegawai, idSatker)
15      jamBaru = toInt(entri.jam_baru)
16
17      JIKA jamBaru > jamSekarang:
18          hasil.pelanggaran += entri.tanggal
19          LANJUTKAN
20      JIKA jamBaru == jamSekarang:
21          hasil.sama++
22          LANJUTKAN
23
24      perbaruiJamLembur(entri.tanggal, jamBaru,
  idPegawai, idSatker)
```



```

25         hasil.diubah++
26
27         tandaiSemuaSebagaiDiperiksa(idSatker, idPegawai,
28         awal, akhir)
29         akhiriTransaksi()
30     KEMBALIKAN hasil
31 END FUNCTION

```

Kode 3.10: Pseudocode penyimpanan koreksi lembur

Fungsi ini memastikan proses koreksi berlangsung aman dan konsisten. Sebagai contoh penggunaan, ketika Kabag menekan tombol Simpan setelah mengubah jam lembur, sistem memanggil fungsi ini untuk memperbarui data dan memperbarui status koreksi secara otomatis.

#### 6. Validasi Akses Satker

Fungsi ini berfungsi untuk memastikan bahwa pengguna hanya dapat mengakses dan mengubah data lembur milik satuan kerja yang menjadi kewenangannya. Dengan cara ini, sistem mencegah kebocoran data antar unit dan menjaga keamanan operasional aplikasi. Proses validasi dilakukan dengan membandingkan identitas satuan kerja pengguna yang sedang login dengan parameter satuan kerja yang dikirim dari URL. Jika keduanya tidak sesuai, sistem akan menolak akses dengan kode 403. Alur validasi dijelaskan pada Pseudocode 3.11.

```

1 FUNCTION validasiAkses(user, idSatkerURL):
2     satkerDiizinkan = user.idbagian ATAU user.idbiro
3     JIKA satkerDiizinkan != idSatkerURL:
4         ERROR 403 "Pengguna tidak memiliki izin untuk
5         mengakses data ini"
6     END FUNCTION

```

Kode 3.11: Pseudocode validasi hak akses satker

Fungsi ini dijalankan sebelum sistem mengambil data lembur apa pun untuk memastikan hanya pengguna yang berwenang yang dapat melihat atau mengubah data. Sebagai contoh penggunaan, setiap kali Kabag membuka halaman koreksi atau detail lembur, sistem menjalankan validasi ini terlebih dahulu sebelum menampilkan data.

### 3.3.5 Modul PIPK

Modul PIPK merupakan bagian dari sistem DIGITall yang mendukung pelaksanaan penilaian pengendalian internal atas pelaporan keuangan di lingkungan Sekretariat Jenderal DPR RI. Modul ini mengintegrasikan proses penetapan tim penilai, penetapan kode akun signifikan, penugasan unit kerja, pengisian matriks pengendalian,

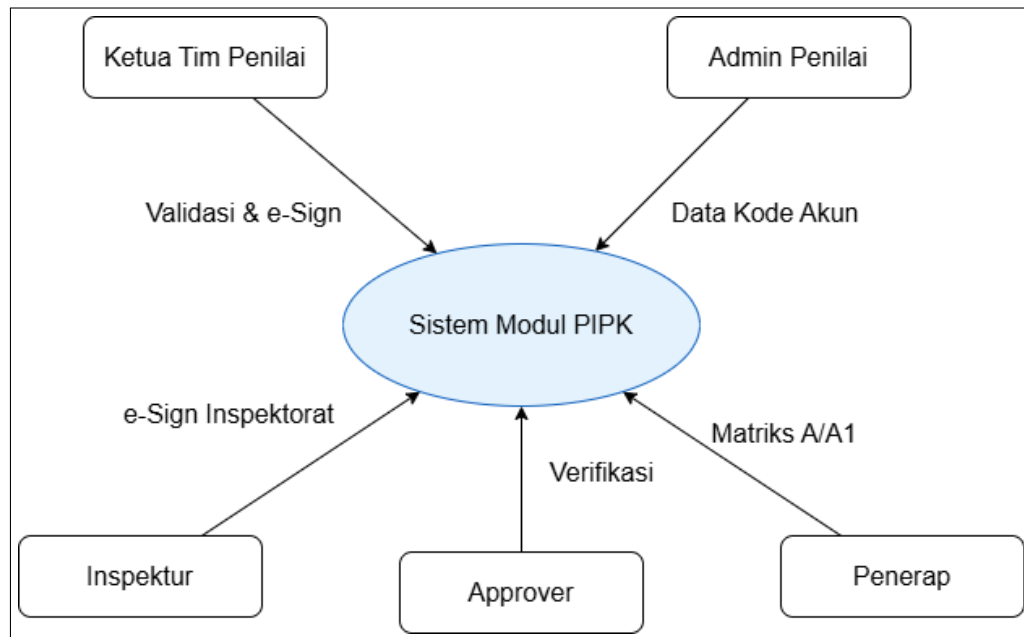
serta verifikasi berjenjang hingga menghasilkan dokumen yang tervalidasi dan dapat diaudit.

#### **A Data Flow Diagram (DFD) Modul PIPK**

Data Flow Diagram (DFD) digunakan untuk menggambarkan alur pergerakan data pada Modul PIPK, mulai dari input data oleh aktor sistem hingga penyimpanan dan pencatatan aktivitas ke dalam basis data. DFD ini membantu menjelaskan hubungan antara proses, entitas eksternal, serta penyimpanan data yang terlibat dalam penilaian pengendalian internal atas pelaporan keuangan. DFD Modul PIPK disajikan dalam dua tingkat, yaitu DFD Level 0 dan DFD Level 1. DFD Level 0 memberikan gambaran umum interaksi sistem dengan aktor eksternal, sedangkan DFD Level 1 menjabarkan proses utama ke dalam subproses yang lebih rinci.

##### **1. DFD Level 0 Modul PIPK**

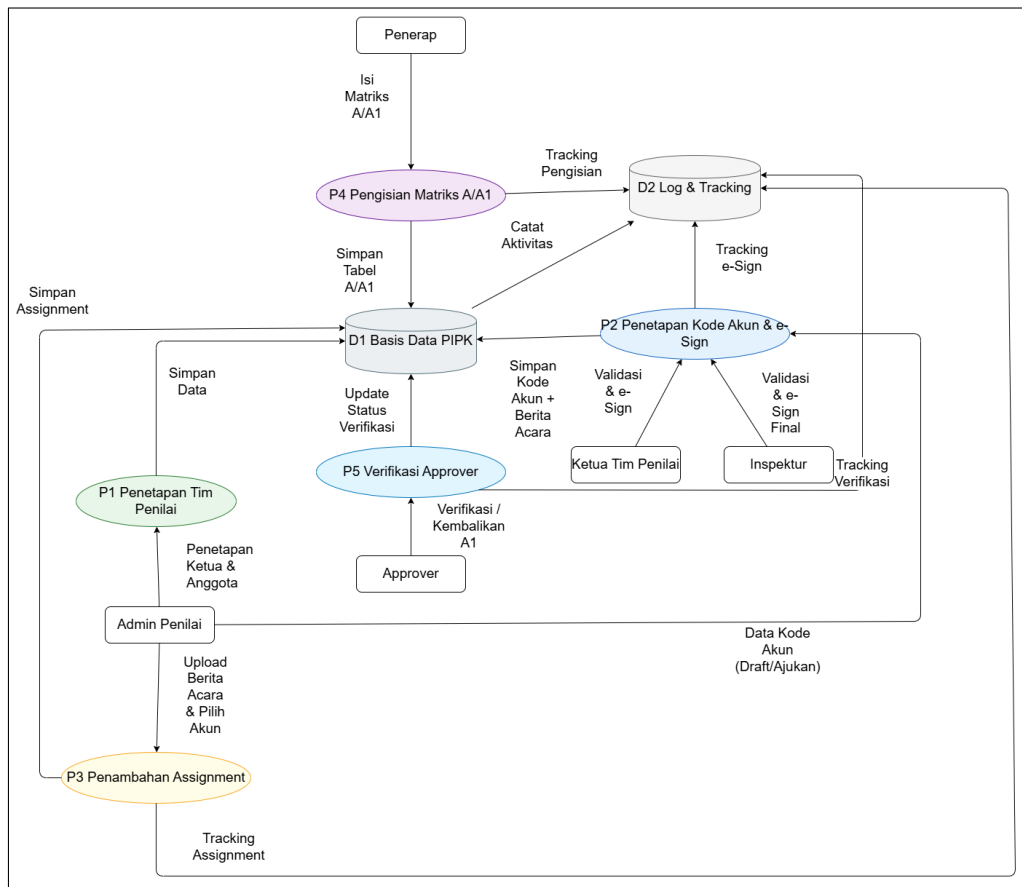
Gambar 3.34 menunjukkan Data Flow Diagram Level 0 Modul PIPK. Pada diagram ini, Sistem Modul PIPK direpresentasikan sebagai satu proses utama yang berinteraksi dengan beberapa entitas eksternal, yaitu Admin Penilai, Penerap, Ketua Tim Penilai, Inspektur, dan Approver. Admin Penilai memberikan input berupa data kode akun dan pengaturan awal penilaian ke dalam sistem. Data tersebut diproses oleh sistem dan disimpan ke dalam basis data sebagai dasar pelaksanaan penilaian. Penerap kemudian menerima data penugasan dari sistem untuk melakukan pengisian matriks pengendalian pada Tabel A. Ketua Tim Penilai dan Inspektur berinteraksi dengan sistem dalam proses validasi serta tanda tangan elektronik (*e-sign*) terhadap kode akun dan dokumen pendukung. Selanjutnya, Approver melakukan verifikasi akhir terhadap hasil pengisian matriks. Seluruh aktivitas dan perubahan status dicatat oleh sistem melalui mekanisme log dan tracking sehingga proses penilaian dapat ditelusuri dan diaudit.



Gambar 3.34. Data Flow Diagram Level 0 Modul PIPK

## 2. DFD Level 1 Modul PIPK

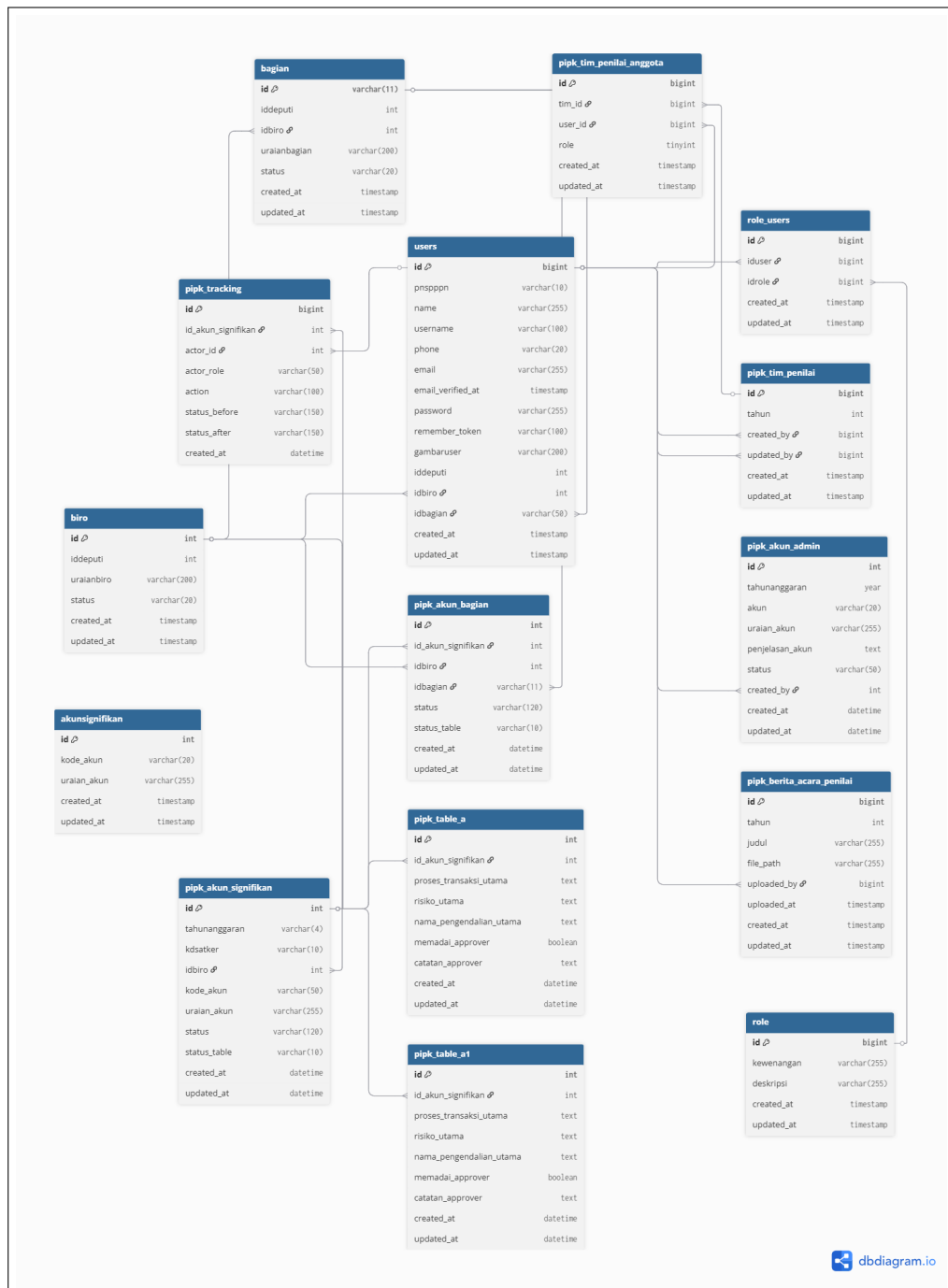
Gambar 3.35 memperlihatkan Data Flow Diagram Level 1 Modul PIPK yang merupakan penguraian lebih rinci dari proses utama pada DFD Level 0. Pada level ini, sistem dipecah menjadi beberapa proses inti yang saling terhubung sesuai alur bisnis Modul PIPK. Proses P1 Penetapan Tim Penilai menggambarkan aktivitas Admin Penilai dalam menetapkan Ketua Tim Penilai dan anggota tim. Data hasil penetapan ini disimpan ke dalam basis data PIPK sebagai dasar kewenangan validasi dan tanda tangan elektronik. Proses P2 Penetapan Kode Akun dan *e-sign* menunjukkan alur pengajuan dan pengesahan kode akun oleh Ketua Tim Penilai dan Inspektur hingga menghasilkan data kode akun yang telah disetujui. Proses P3 Penambahan Assignment menggambarkan penugasan akun ke unit kerja berdasarkan kode akun yang telah disahkan. Data assignment ini kemudian digunakan pada proses P4 Pengisian Matriks A, di mana Penerap melakukan pengisian matriks pengendalian internal sesuai penugasan yang diterima. Hasil pengisian disimpan ke dalam basis data dan status proses diperbarui oleh sistem. Proses terakhir, P5 Verifikasi Approver, menunjukkan tahapan pemeriksaan dan verifikasi hasil pengisian matriks oleh Approver. Pada tahap ini, sistem dapat memperbarui status sebagai terverifikasi atau mengembalikan data untuk dilakukan koreksi. Seluruh aktivitas pada setiap proses dicatat dalam penyimpanan log dan tracking sehingga riwayat penilaian dapat dipantau secara menyeluruh.



Gambar 3.35. Data Flow Diagram Level 1 Modul PIPK

## B Entity Relationship Diagram (ERD) Modul PIPK

Entity Relationship Diagram (ERD) digunakan untuk menggambarkan struktur basis data pada Modul PIPK serta hubungan antar entitas yang mendukung proses penilaian pengendalian internal atas pelaporan keuangan. Sebagaimana ditunjukkan pada Gambar 3.36, ERD Modul PIPK terdiri dari beberapa kelompok tabel, yaitu tabel referensi organisasi, tabel penetapan dan penugasan akun, tabel penilaian, serta tabel pendukung untuk pengelolaan pengguna dan pelacakan aktivitas. Secara umum, hubungan antar tabel pada ERD ini didominasi oleh relasi satu ke banyak (one-to-many) yang mencerminkan karakteristik proses penilaian yang bersifat berjenjang, berulang, dan melibatkan banyak unit kerja serta pengguna.



Gambar 3.36. Entity Relationship Diagram Modul PIPK

Berdasarkan struktur dan relasi yang ditampilkan pada Gambar 3.36, hubungan antar tabel pada ERD Modul PIPK dapat dijelaskan sebagai berikut.

#### 1. Relasi biro dan bagian

Relasi antara tabel biro dan bagian bersifat satu ke banyak, di mana satu biro dapat membawahi lebih dari satu bagian, sedangkan satu bagian hanya berada pada satu biro. Relasi ini mencerminkan struktur organisasi Sekretariat Jenderal DPR

RI yang bersifat hierarkis dan menjadi dasar pemetaan unit kerja dalam proses penilaian PIPK. Tabel biro dan bagian memiliki relasi satu ke banyak dengan tabel users. Satu biro atau bagian dapat memiliki banyak pengguna, sedangkan setiap pengguna hanya terasosiasi pada satu biro dan satu bagian. Relasi ini digunakan untuk mengatur hak akses serta membatasi ruang lingkup data yang dapat diakses oleh masing-masing pengguna sesuai dengan unit kerjanya.

2. Relasi akun admin dan akun signifikan

Relasi antara tabel akun admin dan akun signifikan pada Gambar 3.36 bersifat satu ke banyak. Satu akun yang didefinisikan oleh admin dapat ditetapkan sebagai akun signifikan pada beberapa unit kerja atau pada periode tahun anggaran yang berbeda. Relasi ini memungkinkan pemanfaatan data akun secara berulang tanpa duplikasi dan mendukung konsistensi penilaian antar periode.

3. Relasi akun signifikan dan akun bagian

Sebagaimana ditunjukkan pada Gambar 3.36, relasi antara tabel akun signifikan dan akun bagian bersifat satu ke banyak. Satu akun signifikan dapat ditugaskan ke beberapa unit kerja, baik pada tingkat biro maupun bagian. Relasi ini mendukung mekanisme *assignment* sehingga satu akun keuangan dapat dinilai secara paralel oleh lebih dari satu unit kerja.

4. Relasi akun signifikan dengan tabel penilaian A dan A1

Pada Gambar 3.36 terlihat bahwa tabel akun signifikan memiliki relasi satu ke banyak dengan tabel penilaian A dan tabel penilaian A1. Satu akun signifikan dapat memiliki banyak entri penilaian yang merepresentasikan proses transaksi, risiko, dan pengendalian internal yang berbeda. Tabel penilaian A digunakan untuk pencatatan awal hasil penerapan, sedangkan tabel penilaian A1 digunakan untuk mencatat perbaikan atau penilaian lanjutan.

5. Relasi akun signifikan dan tabel tracking aktivitas

Relasi antara tabel akun signifikan dan tabel tracking pada Gambar 3.36 bersifat satu ke banyak. Setiap akun signifikan dapat memiliki banyak catatan tracking yang merekam perubahan status, tahapan proses, serta aktivitas pengguna selama proses penilaian berlangsung. Relasi ini berfungsi sebagai audit trail untuk menjamin keterlacakan seluruh proses penilaian secara kronologis.

6. Relasi tim penilai dan anggota tim penilai

Gambar 3.36 menunjukkan bahwa relasi antara tabel tim penilai dan tabel anggota tim penilai bersifat satu ke banyak. Satu tim penilai dapat terdiri dari beberapa anggota dengan peran yang berbeda, seperti ketua dan anggota. Relasi ini mendukung mekanisme kerja kolektif dalam proses evaluasi, validasi, dan persetujuan hasil penilaian.

7. Relasi pengguna dengan aktivitas sistem

Pada Gambar 3.36 juga ditunjukkan bahwa tabel users memiliki relasi satu ke banyak dengan tabel tracking. Satu pengguna dapat melakukan banyak aktivitas di dalam sistem, dan setiap aktivitas tersebut dicatat sebagai satu entri *tracking*. Relasi ini memungkinkan sistem mencatat jejak aktivitas pengguna secara terstruktur

untuk mendukung pengawasan, evaluasi, dan akuntabilitas penggunaan sistem.

### C Struktur Basis Data Modul PIPK

Struktur basis data Modul PIPK dirancang untuk mendukung seluruh tahapan proses penilaian pengendalian internal atas pelaporan keuangan secara terintegrasi dan berjenjang. Setiap tabel memiliki fungsi yang saling berkaitan, mulai dari pembentukan tim penilai, penetapan dan pengesahan kode akun, penugasan unit kerja, pengisian matriks pengendalian, hingga proses verifikasi dan pencatatan aktivitas. Rincian struktur dan fungsi masing-masing tabel dijelaskan sebagai berikut.

#### 1. Tabel akunsignifikan

Tabel akunsignifikan berfungsi sebagai tabel referensi akun yang menjadi sumber data awal dalam proses penetapan kode akun. Tabel ini digunakan ketika admin memilih metode penambahan akun dari basis data, sehingga admin tidak perlu memasukkan kode akun secara manual. Data pada tabel ini bersifat master dan tidak langsung terlibat dalam proses penilaian, namun menjadi dasar pembentukan akun pada tahap berikutnya. Struktur tabel akunsignifikan dapat dilihat pada Tabel 3.15.

Tabel 3.15. Struktur tabel akunsignifikan

Atribut	Tipe Data	Keterangan
id	INT	Primary key akun referensi.
kode_akun	VARCHAR(20)	Kode akun keuangan.
uraian_akun	VARCHAR(255)	Uraian akun keuangan.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

#### 2. Tabel biro

Tabel biro menyimpan data unit kerja pada tingkat biro yang digunakan sebagai referensi struktur organisasi dalam Modul PIPK. Data biro digunakan saat proses penambahan assignment untuk menentukan unit kerja penanggung jawab akun signifikan. Keberadaan tabel ini memungkinkan sistem memetakan akun pada tingkat organisasi yang lebih luas sebelum diturunkan ke tingkat bagian. Struktur tabel biro disajikan pada Tabel 3.16.

Tabel 3.16. Struktur tabel biro

Atribut	Tipe Data	Keterangan
id	INT	Primary key biro.
iddeputi	INT	Referensi deputi.
uraianbiro	VARCHAR(200)	Nama biro.
status	SET	Status aktif atau tidak aktif.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

### 3. Tabel bagian

Tabel bagian menyimpan data unit kerja pada tingkat bagian yang berada di bawah biro. Tabel ini digunakan untuk membatasi cakupan *assignment*, sehingga akun signifikan dapat ditugaskan ke seluruh biro atau hanya ke bagian tertentu. Dengan tabel ini, sistem dapat menampilkan akun hanya kepada penerap yang sesuai dengan kewenangannya. Struktur tabel bagian ditunjukkan pada Tabel 3.17.

Tabel 3.17. Struktur tabel bagian

Atribut	Tipe Data	Keterangan
id	VARCHAR(11)	Primary key bagian.
iddeputi	INT	Referensi deputi.
idbiro	INT	Referensi biro.
uraianbagian	VARCHAR(200)	Nama bagian.
status	SET	Status aktif atau tidak aktif.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

### 4. Tabel users

Tabel users menyimpan data seluruh pengguna sistem yang terlibat dalam Modul PIPK, termasuk admin penilai, penerap, approver, dan penilai. Informasi unit kerja pada tabel ini digunakan untuk menentukan akun mana yang dapat diakses oleh pengguna. Dengan demikian, tabel ini berperan penting dalam pengendalian akses data berdasarkan biro dan bagian. Struktur tabel users dapat dilihat pada Tabel 3.18.



Tabel 3.18. Struktur tabel users

Atribut	Tipe Data	Keterangan
id	bigint (unsigned)	Primary key, AUTO_INCREMENT.
pnspppn	varchar(10)	Identitas status kepegawaian (mis. PNS/PPPK/CPNS) sesuai kebutuhan aplikasi.
name	varchar(255)	Nama pengguna.
username	varchar(100)	Nama akun pengguna (opsional).
phone	varchar(20)	Nomor telepon pengguna (opsional).
email	varchar(255)	Email pengguna (unik/untuk autentikasi).
email_verified_at	timestamp	Waktu verifikasi email (opsional).
password	varchar(255)	Kata sandi terenkripsi.
remember_token	varchar(100)	Token sesi <i>remember me</i> (opsional).
gambaruser	varchar(200)	Path/nama file foto pengguna (opsional).
iddeputi	int	Kode deputi pengguna (opsional).
idbiro	int	Kode biro pengguna (opsional).
idbagian	varchar(50)	Kode bagian pengguna (opsional).
created_at	timestamp	Waktu data dibuat.
updated_at	timestamp	Waktu data diperbarui.

##### 5. Tabel role

Tabel role menyimpan daftar peran atau kewenangan pengguna dalam sistem. Peran ini menentukan fungsi apa saja yang dapat dijalankan oleh pengguna, seperti melakukan input akun, mengisi matriks, atau melakukan verifikasi. Tabel ini mendukung penerapan kontrol akses berbasis peran. Struktur tabel role disajikan pada Tabel 3.19.

Tabel 3.19. Struktur tabel role

Atribut	Tipe Data	Keterangan
id	BIGINT	Primary key peran.
kewenangan	VARCHAR(255)	Nama peran pengguna.
deskripsi	VARCHAR(255)	Penjelasan peran.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

6. Tabel role\_users

Tabel role\_users digunakan untuk menghubungkan pengguna dengan peran yang dimiliki. Dengan tabel ini, satu pengguna dapat memiliki satu atau lebih peran sesuai kebutuhan sistem. Relasi ini memungkinkan fleksibilitas dalam pengaturan hak akses tanpa mengubah struktur pengguna. Struktur tabel role\_users ditampilkan pada Tabel 3.20.

Tabel 3.20. Struktur tabel role\_users

Atribut	Tipe Data	Keterangan
id	BIGINT	Primary key relasi.
iduser	BIGINT	Referensi pengguna.
idrole	BIGINT	Referensi peran.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

7. Tabel pipk\_tim\_penilai

Tabel pipk\_tim\_penilai menyimpan data pembentukan tim penilai pada setiap tahun penilaian. Pembentukan tim ini menjadi tahap awal karena menentukan siapa yang memiliki kewenangan melakukan validasi dan tanda tangan elektronik dalam proses penetapan kode akun. Struktur tabel pipk\_tim\_penilai dapat dilihat pada Tabel 3.21.

Tabel 3.21. Struktur tabel pipk\_tim\_penilai

Atribut	Tipe Data	Keterangan
id	BIGINT	Primary key tim penilai.
tahun	INT	Tahun penilaian.
created_by	BIGINT	Pengguna pembuat tim.
updated_by	BIGINT	Pengguna pengubah tim.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

8. Tabel pipk\_tim\_penilai\_anggota

Tabel pipk\_tim\_penilai\_anggota menyimpan daftar anggota dalam satu tim penilai beserta perannya. Data pada tabel ini digunakan untuk membedakan ketua tim

dan anggota tim, di mana ketua tim memiliki kewenangan e-sign dalam proses penetapan kode akun. Struktur tabel ini ditunjukkan pada Tabel 3.22.

Tabel 3.22. Struktur tabel pipk\_tim\_penilai\_anggota

Atribut	Tipe Data	Keterangan
id	BIGINT	Primary key anggota tim.
tim_id	BIGINT	Referensi tim penilai.
user_id	BIGINT	Referensi pengguna.
role	TINYINT	Peran ketua atau anggota.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

#### 9. Tabel pipk\_akun\_admin

Tabel pipk\_akun\_admin merupakan tabel utama dalam proses penetapan kode akun. Seluruh aktivitas mulai dari input akun, penyimpanan draft, pengajuan, penolakan sebagian atau seluruh akun, hingga pengesahan oleh Inspektorat dilakukan melalui perubahan status pada tabel ini. Struktur tabel pipk\_akun\_admin disajikan pada Tabel 3.23.

Tabel 3.23. Struktur tabel pipk\_akun\_admin

Atribut	Tipe Data	Keterangan
id	INT	Primary key akun admin.
tahunanggaran	YEAR	Tahun anggaran.
akun	VARCHAR(20)	Kode akun.
uraian_akun	VARCHAR(255)	Uraian akun.
penjelasan_akun	TEXT	Penjelasan akun.
status	ENUM	Status proses penetapan.
created_by	INT	Admin pembuat data.
created_at	DATETIME	Waktu pembuatan data.
updated_at	DATETIME	Waktu pembaruan data.

#### 10. Tabel pipk\_berita\_acara\_penilai

Tabel pipk\_berita\_acara\_penilai menyimpan dokumen berita acara hasil penetapan kode akun yang telah disahkan. Dokumen ini menjadi prasyarat dalam proses penambahan *assignment* karena hanya akun yang tercantum dalam berita acara yang dapat ditugaskan. Struktur tabel ini ditunjukkan pada Tabel 3.24.

Tabel 3.24. Struktur tabel pipk\_berita\_acara\_penilai

Atribut	Tipe Data	Keterangan
id	BIGINT	Primary key berita acara.
tahun	INT	Tahun penilaian.
judul	VARCHAR(255)	Judul dokumen.
file_path	VARCHAR(255)	Lokasi file.
uploaded_by	BIGINT	Pengunggah dokumen.
uploaded_at	TIMESTAMP	Waktu unggah.
created_at	TIMESTAMP	Waktu pembuatan data.
updated_at	TIMESTAMP	Waktu pembaruan data.

11. Tabel pipk\_akun\_signifikan

Tabel pipk\_akun\_signifikan menyimpan akun yang telah disahkan dan ditugaskan kepada unit kerja. Tabel ini menjadi pusat alur penilaian karena menjadi referensi utama bagi tabel assignment, tabel pengisian matriks, dan tabel tracking. Struktur tabel ini disajikan pada Tabel 3.25.

Tabel 3.25. Struktur tabel pipk\_akun\_signifikan

Atribut	Tipe Data	Keterangan
id	INT	Primary key akun signifikan.
tahunanggaran	VARCHAR(4)	Tahun penilaian.
kdsatker	VARCHAR(10)	Kode satuan kerja.
idbiro	INT	Biro penanggung jawab.
kode_akun	VARCHAR(50)	Kode akun signifikan.
uraian_akun	VARCHAR(255)	Uraian akun.
status	VARCHAR(120)	Status proses penilaian.
status_table	ENUM	Tahap A atau A1.
created_at	DATETIME	Waktu pembuatan data.
updated_at	DATETIME	Waktu pembaruan data.

12. Tabel pipk\_akun\_bagian

Tabel pipk\_akun\_bagian mengatur cakupan assignment akun signifikan terhadap unit kerja. Dengan tabel ini, sistem dapat menentukan apakah akun berlaku untuk seluruh bagian dalam satu biro atau hanya bagian tertentu. Struktur tabel ini dapat dilihat pada Tabel 3.26.

Tabel 3.26. Struktur tabel pipk\_akun\_bagian

Atribut	Tipe Data	Keterangan
id	INT	Primary key pemetaan.
id_akun_signifikan	INT	Referensi akun signifikan.
idbiro	INT	Referensi biro.
idbagian	INT	Referensi bagian (opsional).
status	VARCHAR(120)	Status assignment.
status_table	ENUM	Tahap penilaian.
created_at	DATETIME	Waktu pembuatan data.
updated_at	DATETIME	Waktu pembaruan data.

13. Tabel pipk\_table\_a

Tabel pipk\_table\_a menyimpan hasil pengisian awal matriks pengendalian oleh penerap. Data pada tabel ini menjadi objek utama pemeriksaan oleh approver. Struktur tabel ini disajikan pada Tabel 3.27.

Tabel 3.27. Struktur tabel pipk\_table\_a

Atribut	Tipe Data	Keterangan
id	INT	Primary key baris.
id_akun_signifikan	INT	Referensi akun signifikan.
proses_transaksi_utama	TEXT	Proses transaksi.
risiko_utama	TEXT	Risiko utama.
nama_pengendalian_utama	TEXT	Pengendalian utama.
memadai_approver	BOOLEAN	Status memadai.
catatan_approver	TEXT	Catatan verifikasi.
created_at	DATETIME	Waktu pembuatan data.
updated_at	DATETIME	Waktu pembaruan data.

14. Tabel pipk\_table\_a1

Tabel pipk\_table\_a1 menyimpan hasil perbaikan atas baris penilaian yang dinyatakan tidak memadai. Tabel ini digunakan saat terjadi proses koreksi berulang sampai data dinyatakan valid. Struktur tabel ini dapat dilihat pada Tabel 3.28.

Tabel 3.28. Struktur tabel pipk\_table\_a1

Atribut	Tipe Data	Keterangan
id	INT	Primary key baris koreksi.
id_akun_signifikan	INT	Referensi akun signifikan.
proses_transaksi_utama	TEXT	Proses transaksi koreksi.
risiko_utama	TEXT	Risiko koreksi.
nama_pengendalian_utama	TEXT	Pengendalian koreksi.
memadai_approver	BOOLEAN	Status verifikasi.
catatan_approver	TEXT	Catatan koreksi.
created_at	DATETIME	Waktu pembuatan data.
updated_at	DATETIME	Waktu pembaruan data.

#### 15. Tabel pipk\_tracking

Tabel pipk\_tracking digunakan untuk mencatat seluruh aktivitas dan perubahan status dalam Modul PIPK. Tabel ini berfungsi sebagai audit trail sehingga seluruh proses penilaian dapat ditelusuri secara kronologis. Struktur tabel ini disajikan pada Tabel 3.29.

Tabel 3.29. Struktur tabel pipk\_tracking

Atribut	Tipe Data	Keterangan
id	BIGINT	Primary key tracking.
id_akun_signifikan	INT	Referensi akun signifikan.
actor_id	INT	Pengguna yang melakukan aksi.
actor_role	VARCHAR(50)	Peran pengguna.
action	VARCHAR(100)	Jenis aksi.
status_before	VARCHAR(150)	Status sebelum aksi.
status_after	VARCHAR(150)	Status setelah aksi.
created_at	DATETIME	Waktu pencatatan aktivitas.

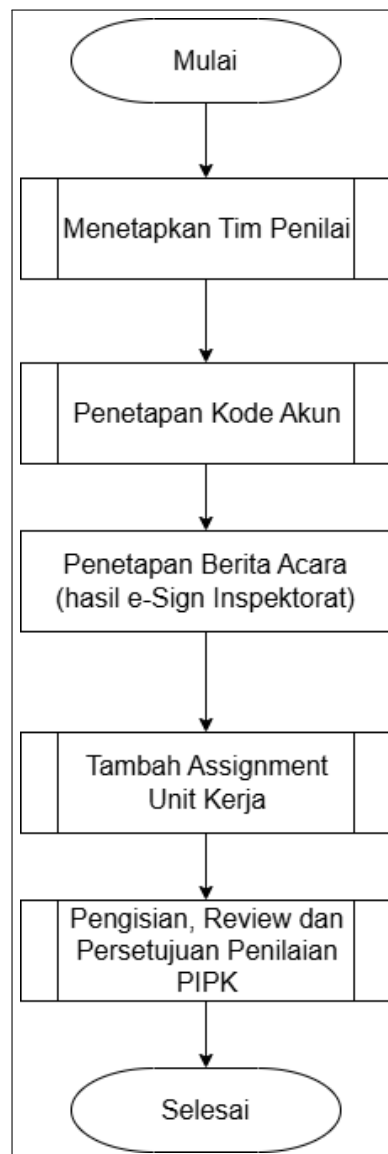
### D Alur Proses Modul PIPK

Alur proses pada Modul PIPK menggambarkan tahapan kerja sistem dalam mendukung penilaian pengendalian internal atas pelaporan keuangan. Alur ini mencakup proses penetapan tim penilai, penetapan kode akun, penetapan berita acara hasil *e-sign* inspektorat, penugasan akun ke unit kerja, serta proses pengisian, *review*, dan persetujuan penilaian PIPK. Setiap tahapan disusun secara berurutan dan saling terhubung untuk memastikan proses penilaian berjalan secara terkontrol dan terdokumentasi.

#### 1. Alur umum proses Modul PIPK

Gambar 3.37 menampilkan gambaran umum alur proses Modul PIPK. Proses dimulai dari penetapan tim penilai oleh admin penilai, dilanjutkan dengan penetapan kode akun yang akan dinilai. Setelah kode akun ditetapkan dan

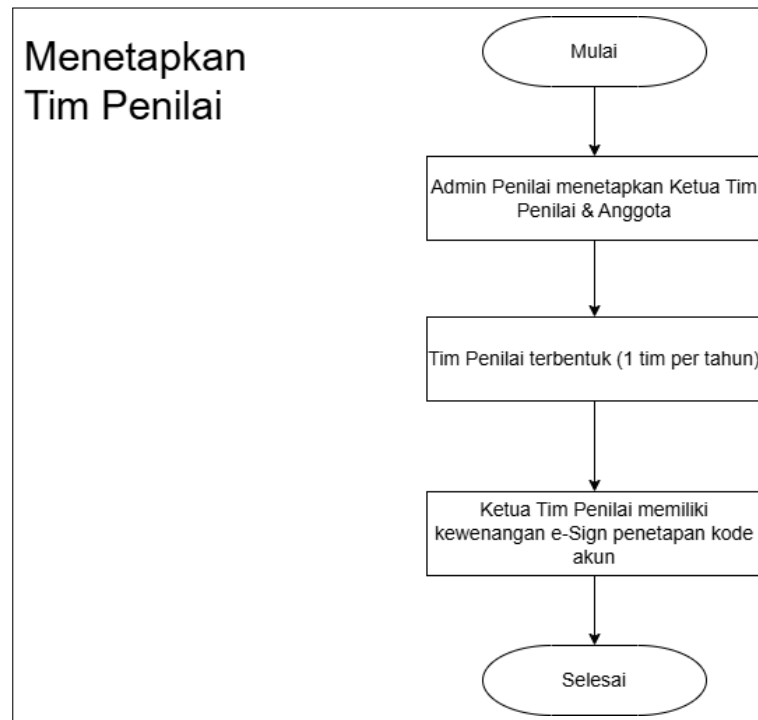
disahkan melalui mekanisme *e-sign* oleh ketua tim penilai dan inspektorat, sistem memungkinkan admin untuk menambahkan assignment unit kerja terhadap akun signifikan. Tahapan selanjutnya adalah proses pengisian, *review*, dan persetujuan penilaian PIPK hingga seluruh tahapan *e-sign* selesai dan proses dinyatakan berakhir.



Gambar 3.37. Flowchart alur umum proses Modul PIPK

## 2. Proses penetapan tim penilai

Gambar 3.38 menggambarkan alur proses penetapan tim penilai dalam Modul PIPK. Proses dimulai ketika admin penilai menetapkan ketua tim penilai beserta anggota tim penilai. Sistem memastikan bahwa dalam satu tahun anggaran hanya terdapat satu tim penilai yang aktif. Setelah tim terbentuk, ketua tim penilai diberikan kewenangan untuk melakukan *e-sign* pada proses penetapan kode akun.

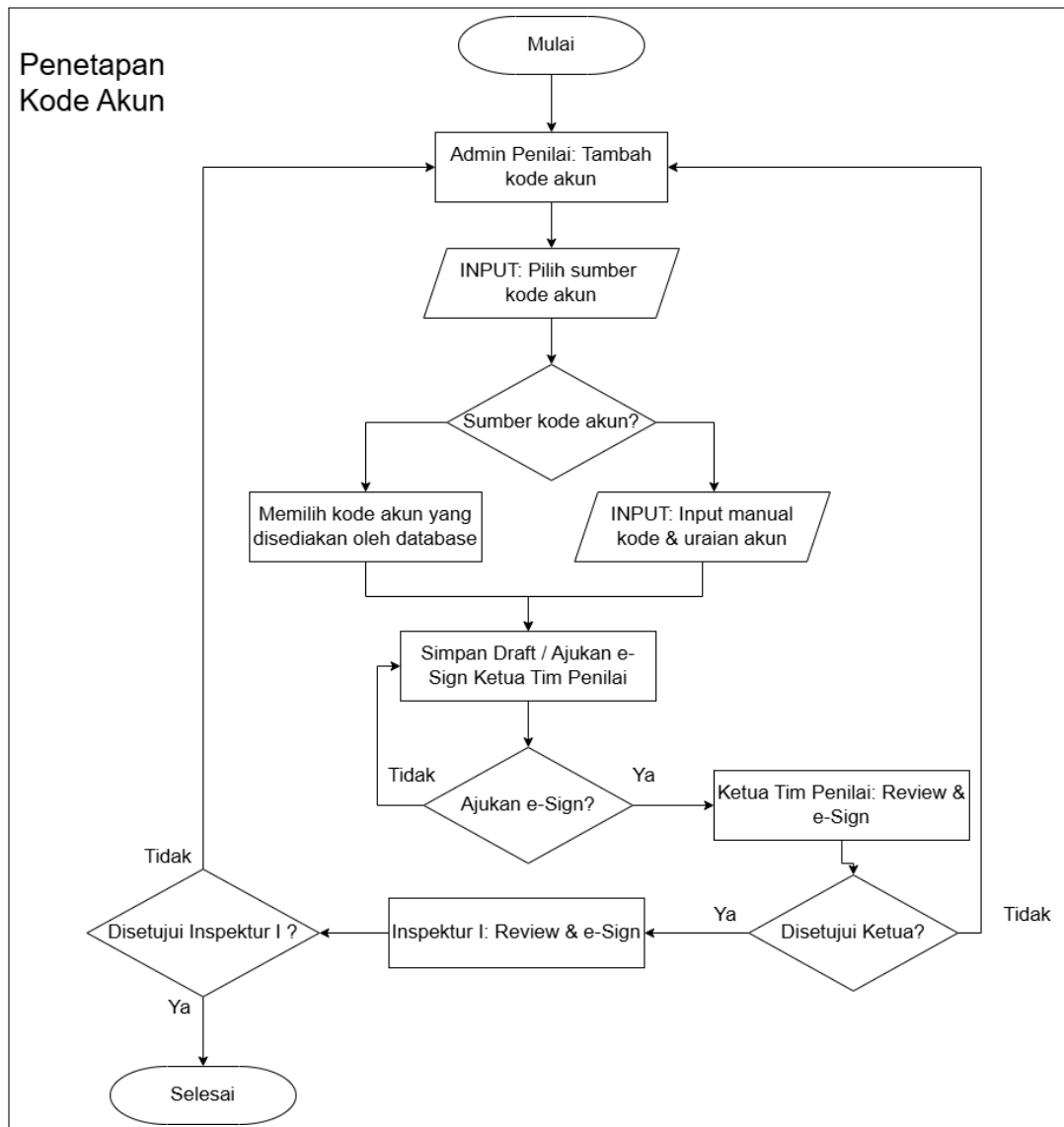


Gambar 3.38. Flowchart proses penetapan tim penilai

### 3. Proses penetapan kode akun

Gambar 3.39 menunjukkan tahapan penetapan kode akun yang akan dinilai dalam PIPK. Admin penilai dapat menambahkan kode akun dengan memilih sumber data kode akun, baik dari database yang tersedia maupun dengan melakukan input manual kode dan uraian akun. Kode akun yang telah ditambahkan dapat disimpan sebagai draft atau diajukan untuk proses *e-sign*. Proses persetujuan dilakukan secara berjenjang, dimulai dari ketua tim penilai dan dilanjutkan oleh inspektur. Apabila kode akun disetujui, maka proses penetapan kode akun dinyatakan selesai dan siap digunakan pada tahap selanjutnya.

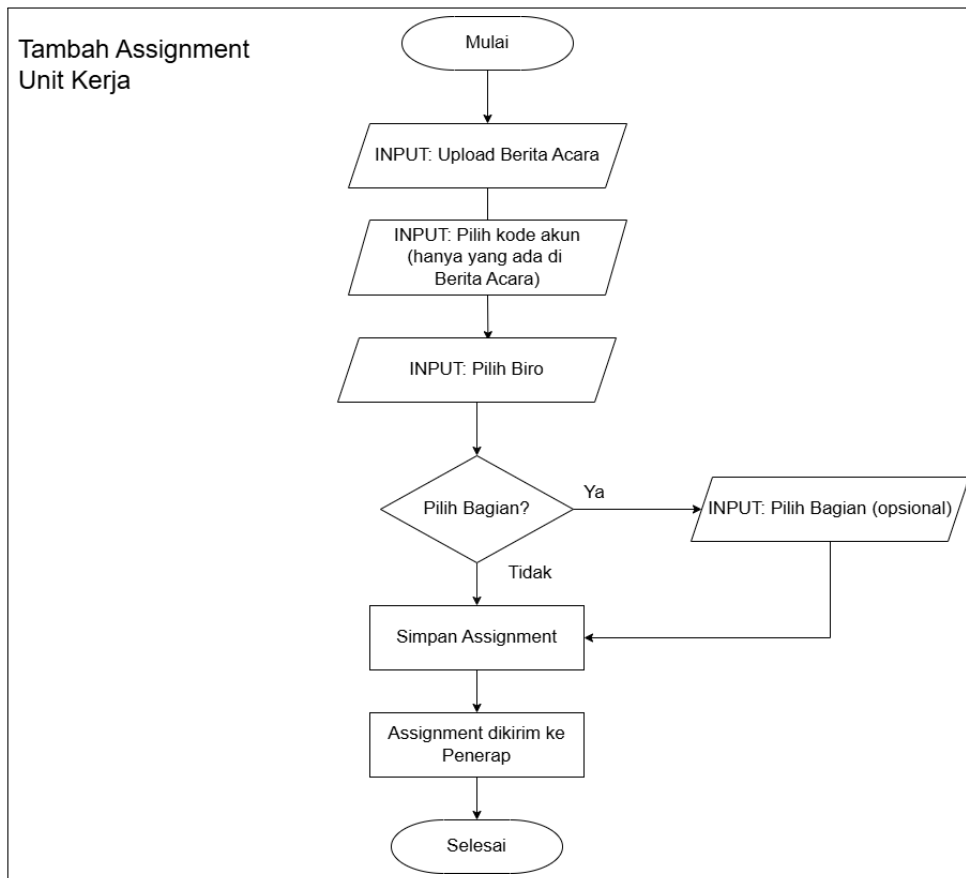




Gambar 3.39. Flowchart proses penetapan kode akun

#### 4. Proses penambahan *assignment* unit kerja

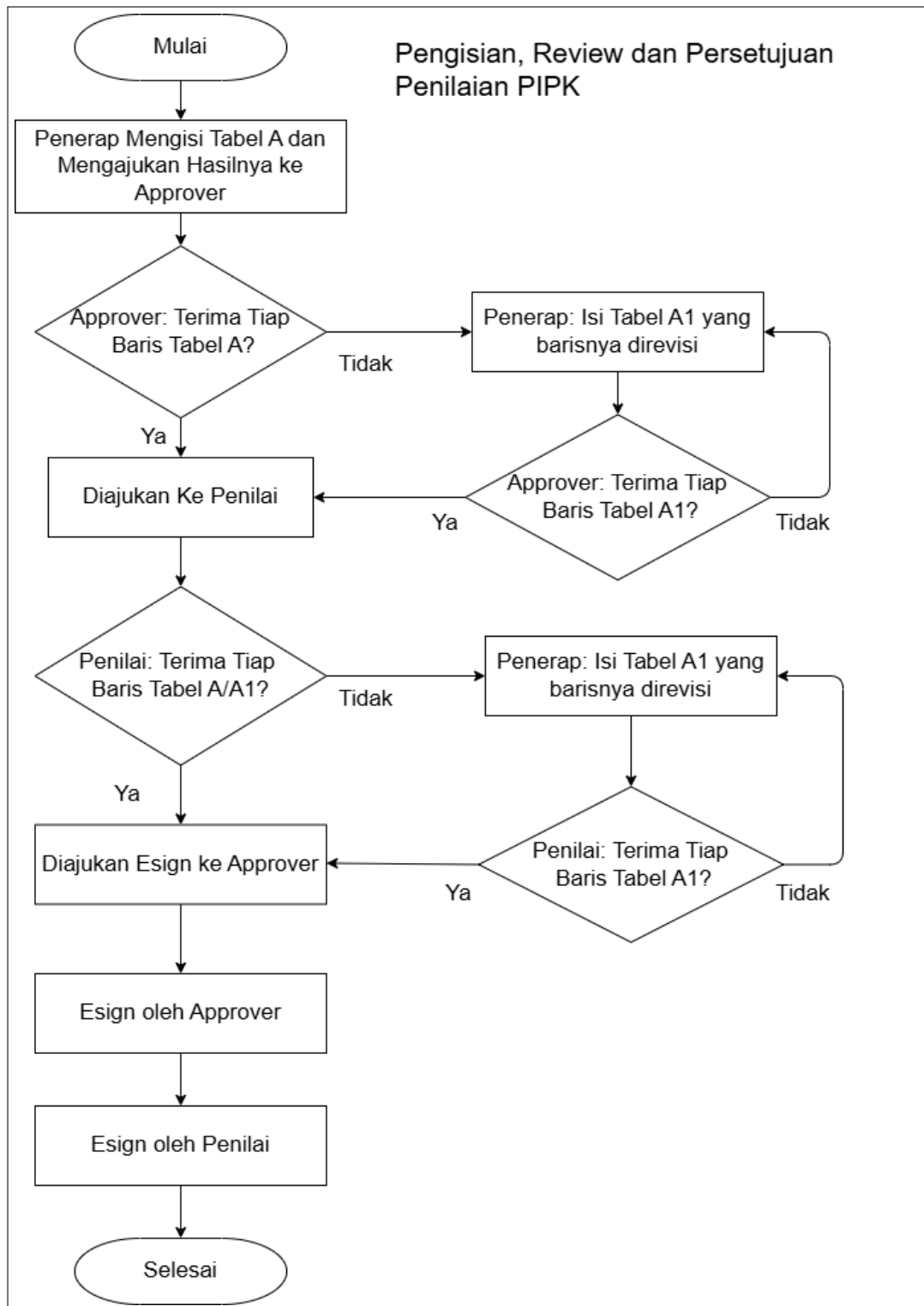
Gambar 3.40 menggambarkan alur penambahan *assignment* unit kerja terhadap akun signifikan. Admin melakukan upload berita acara, memilih kode akun yang tercantum dalam berita acara tersebut, serta menentukan biro dan bagian yang akan diberikan *assignment*. Pemilihan bagian bersifat opsional. Setelah data *assignment* disimpan, sistem secara otomatis mengirimkan *assignment* tersebut kepada penerap untuk ditindaklanjuti pada tahap penilaian.



Gambar 3.40. Flowchart proses penambahan assignment unit kerja

##### 5. Subproses pengisian, *review*, dan persetujuan penilaian PIPK

Gambar 3.41 memperlihatkan alur detail subproses pengisian dan validasi penilaian PIPK. Proses dimulai ketika penerap mengisi Tabel A dan mengajukan hasilnya kepada approver. Apabila terdapat baris penilaian yang belum disetujui, penerap melakukan revisi melalui Tabel A1. Setelah seluruh baris diterima oleh approver, data diajukan kepada penilai untuk dilakukan *review*. Jika penilai menyetujui seluruh hasil penilaian, proses dilanjutkan ke tahap *e-sign* oleh approver dan penilai. Subproses ini berakhir ketika seluruh tahapan *e-sign* selesai.



Gambar 3.41. Flowchart subproses pengisian, review, dan persetujuan penilaian PIPK

## E Perancangan *Wireframe* Modul PIPK

Perancangan *wireframe* pada Modul PIPK bertujuan untuk menggambarkan rancangan antarmuka sistem secara konseptual sebelum tahap implementasi. *Wireframe* disusun dalam bentuk *low-fidelity* untuk memfokuskan perancangan pada struktur halaman, alur interaksi pengguna, serta pembagian fungsi utama tanpa dipengaruhi aspek visual akhir. *Wireframe* Modul PIPK dirancang dalam tujuh halaman utama yang mewakili seluruh proses bisnis sistem, mulai dari penetapan tim penilai, penetapan kode akun, penugasan akun ke unit kerja, pengisian matriks pengendalian, hingga proses verifikasi dan persetujuan. Setiap *wireframe* merepresentasikan satu kelompok fungsi utama agar alur sistem dapat dipahami secara utuh dan terstruktur.

### 1. *Wireframe* Halaman Penetapan Tim Penilai PIPK

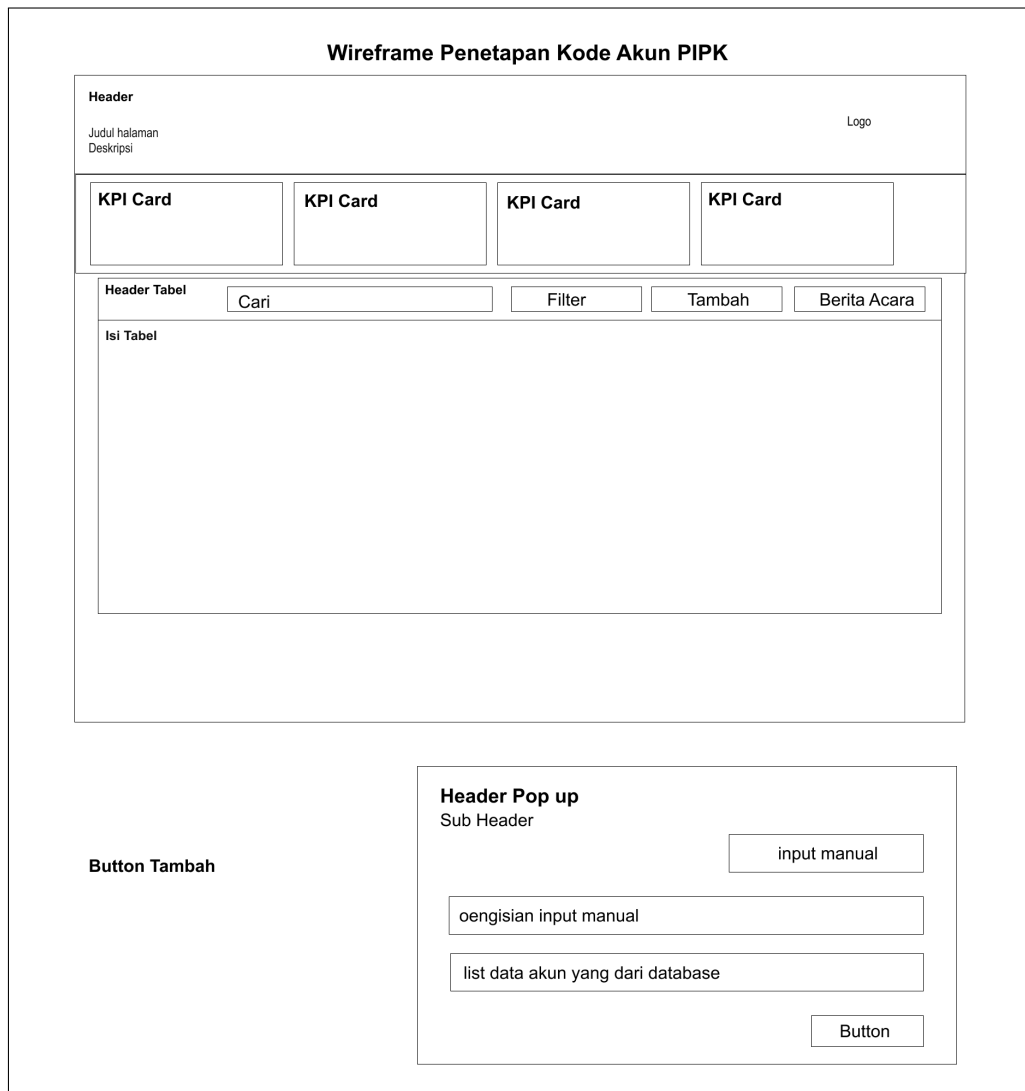
*Wireframe* ini menggambarkan halaman penetapan Ketua Tim Penilai dan anggota tim penilai pada tahun anggaran tertentu. Halaman ini terdiri atas komponen pencarian, filter tahun, tabel daftar penetapan tim, serta tombol untuk menambahkan penetapan baru. Selain itu, disediakan jendela *pop-up* untuk memasukkan data tahun, memilih ketua tim, dan memilih anggota tim penilai. *Wireframe* ini menjadi tahap awal dalam sistem karena penetapan tim penilai diperlukan untuk memberikan kewenangan validasi dan tanda tangan elektronik pada proses selanjutnya. Rancangan *wireframe* halaman penetapan tim penilai ditunjukkan pada Gambar 3.42.



Gambar 3.42. *Wireframe* Halaman Penetapan Tim Penilai PIPK

## 2. *Wireframe* Halaman Penetapan Kode Akun PIPK

*Wireframe* ini merepresentasikan halaman admin penilai dalam mengelola kode akun PIPK. Halaman terdiri atas bagian header, kartu ringkasan informasi, serta tabel daftar kode akun yang telah ditambahkan. Admin dapat melakukan pencarian, penyaringan data, penambahan akun, serta pengelolaan Berita Acara. Pada *wireframe* ini juga ditampilkan jendela *pop-up* untuk menambahkan akun secara manual maupun memilih akun dari basis data. *Wireframe* ini mencerminkan proses penetapan akun dari tahap draft hingga siap diajukan untuk proses validasi. Rancangan *wireframe* halaman penetapan kode akun ditampilkan pada Gambar 3.43.



Gambar 3.43. *Wireframe* Halaman Penetapan Kode Akun PIPK

### 3. *Wireframe* Halaman Daftar Pengajuan

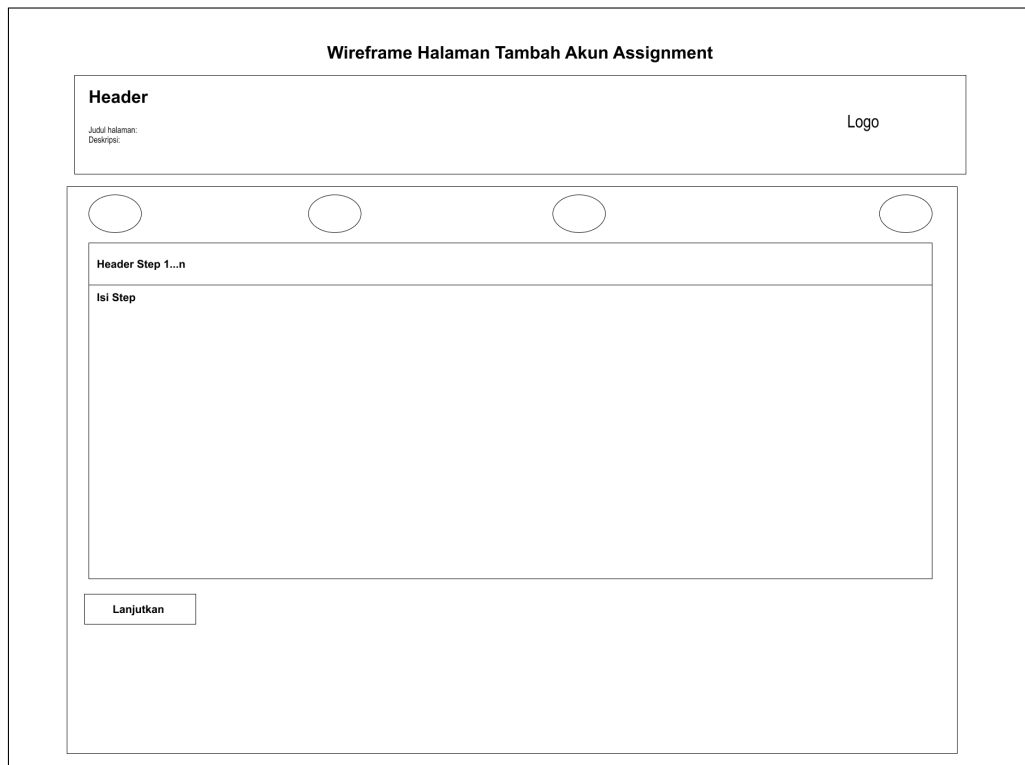
*Wireframe* halaman daftar pengajuan digunakan untuk menampilkan seluruh pengajuan kode akun yang sedang dalam proses validasi. Halaman ini memuat komponen pencarian, filter status pengajuan, tombol akses Berita Acara, serta tabel daftar pengajuan. *Wireframe* ini digunakan oleh pihak penilai dan inspektorat untuk memantau status pengajuan sebelum dilakukan proses persetujuan atau penolakan. Rancangan *wireframe* halaman daftar pengajuan ditunjukkan pada Gambar 3.44.



Gambar 3.44. *Wireframe* Halaman Daftar Pengajuan

#### 4. *Wireframe* Halaman Tambah Akun *Assignment*

*Wireframe* ini menggambarkan proses penugasan akun kepada unit kerja melalui mekanisme bertahap. Halaman dirancang dalam bentuk *wizard* yang terdiri atas beberapa langkah, yaitu unggah Berita Acara, pemilihan kode akun yang telah disahkan, pemilihan biro, serta pemilihan bagian. Struktur bertahap ini bertujuan untuk memastikan bahwa hanya akun yang telah ditandatangani oleh inspektorat yang dapat ditugaskan kepada unit kerja. Rancangan *wireframe* halaman tambah akun *assignment* ditampilkan pada Gambar 3.45.

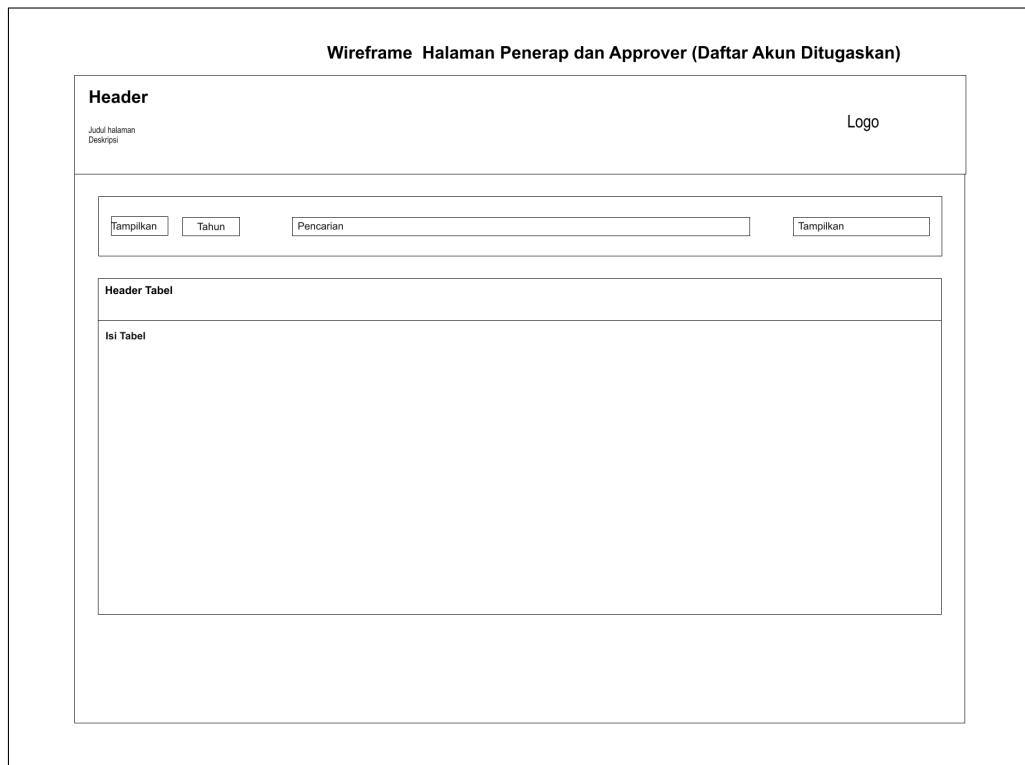


Gambar 3.45. *Wireframe* Halaman Tambah Akun Assignment

5. *Wireframe* Halaman Penerap dan Approver (Daftar Akun Ditugaskan)

*Wireframe* ini digunakan oleh penerap dan approver untuk melihat daftar akun yang telah ditugaskan kepada unit kerja. Halaman ini memuat filter tahun, kolom pencarian, serta tabel daftar akun beserta status pengisiannya. Dari halaman ini, penerap dapat melanjutkan ke pengisian matriks, sedangkan approver dapat memantau akun yang siap diverifikasi. Rancangan *wireframe* halaman daftar akun ditugaskan ditunjukkan pada Gambar 3.46.

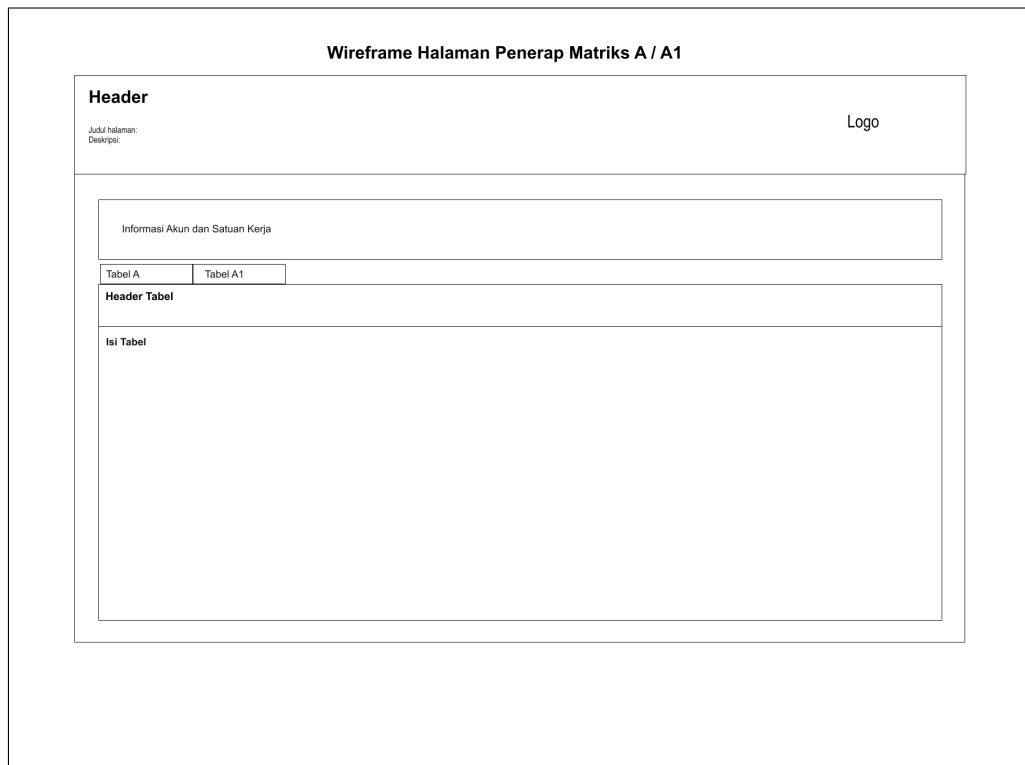




Gambar 3.46. *Wireframe* Halaman Daftar Akun Ditugaskan

#### 6. *Wireframe* Halaman Penerap Matriks A dan A1

*Wireframe* ini menggambarkan halaman pengisian matriks pengendalian oleh penerap. Halaman terdiri atas informasi akun dan satuan kerja, tab pemisah antara Tabel A dan Tabel A1, header tabel, serta area isian matriks. *Wireframe* ini mendukung mekanisme revisi, di mana data yang dinyatakan belum memadai akan dialihkan ke Tabel A1 untuk dilakukan perbaikan sebelum diajukan kembali. Rancangan *wireframe* halaman pengisian matriks ditunjukkan pada Gambar 3.47.



Gambar 3.47. *Wireframe* Halaman Penerap Matriks A dan A1

#### 7. *Wireframe* Halaman *E-Sign Approver*

*Wireframe* ini merepresentasikan halaman persetujuan dan tanda tangan elektronik oleh approver. Halaman menampilkan informasi akun dan satuan kerja, pratinjau dokumen Berita Acara dalam bentuk PDF, serta tombol persetujuan. *Wireframe* ini menegaskan peran approver dalam proses verifikasi sebelum data dilanjutkan ke tahap berikutnya. Rancangan *wireframe* halaman *e-sign* approver ditampilkan pada Gambar 3.48.



Gambar 3.48. Wireframe Halaman E-Sign Approver

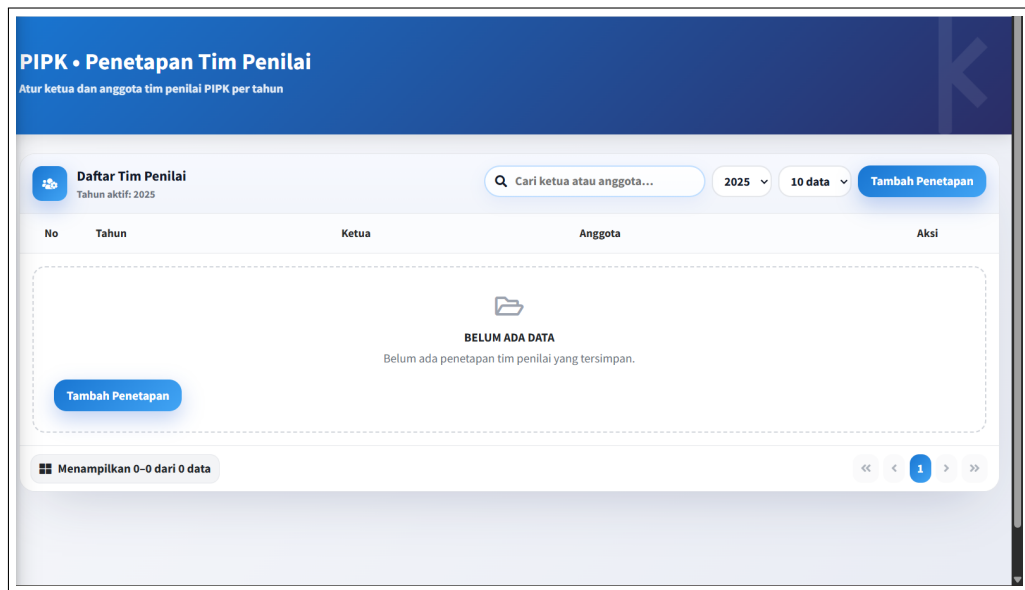
Perancangan *wireframe* pada Modul PIPK dibatasi hanya pada tujuh halaman utama karena seluruh proses bisnis sistem dapat direpresentasikan secara efektif melalui pengelompokan fungsi tersebut. Beberapa halaman operasional yang memiliki struktur dan pola interaksi serupa digabungkan ke dalam satu *wireframe* untuk menghindari redundansi.

## F Implementasi Antarmuka Modul PIPK

Implementasi antarmuka Modul PIPK dirancang untuk mendukung alur kerja penilaian pengendalian internal atas pelaporan keuangan secara berjenjang dan terdokumentasi. Setiap halaman antarmuka disusun mengikuti urutan proses bisnis, mulai dari penetapan tim penilai, penetapan kode akun signifikan, penandatanganan elektronik, hingga pelaksanaan dan verifikasi matriks penilaian oleh unit kerja terkait.

### 1. Halaman penetapan tim penilai

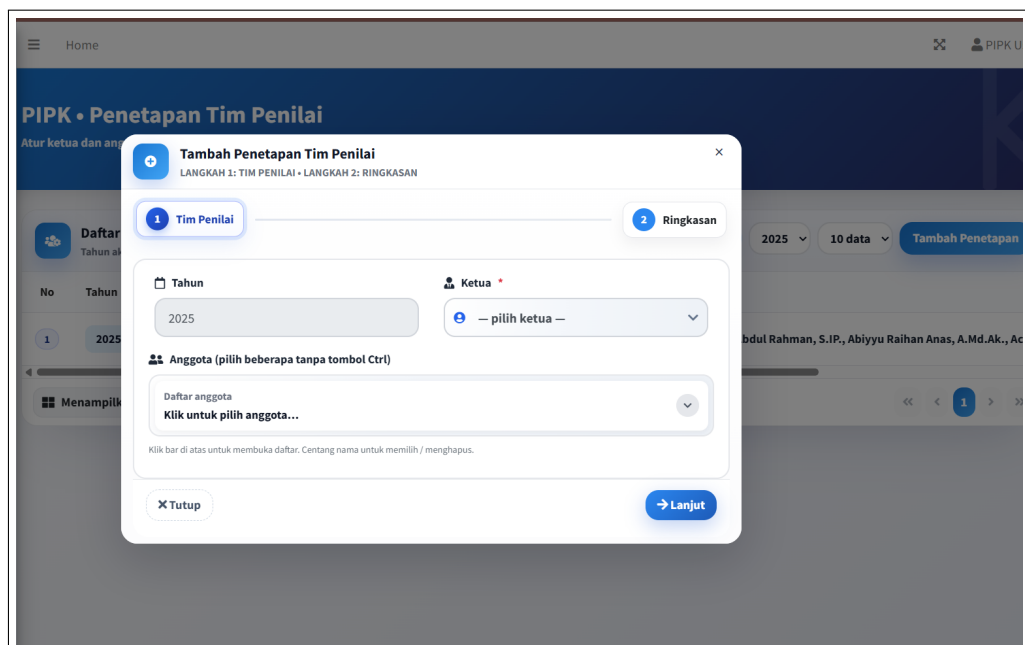
Gambar 3.49 menampilkan halaman utama penetapan tim penilai PIPK. Halaman ini digunakan untuk mengelola daftar tim penilai per tahun anggaran. Informasi yang ditampilkan meliputi tahun aktif, nama ketua tim penilai, serta daftar anggota tim. Keberadaan halaman ini menjadi prasyarat utama karena ketua tim penilai akan memperoleh kewenangan untuk melakukan penandatanganan elektronik pada tahap penetapan kode akun.



Gambar 3.49. Halaman penetapan tim penilai PIPK

## 2. Form tambah penetapan tim penilai

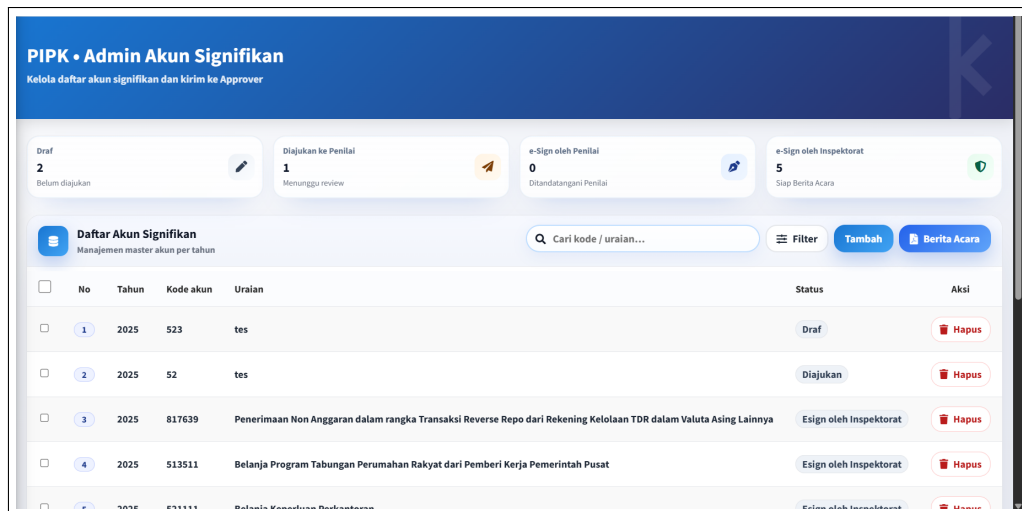
Gambar 3.50 menunjukkan formulir penambahan tim penilai. Pengguna memilih tahun penilaian, menetapkan satu orang sebagai ketua, serta memilih beberapa anggota tim. Struktur formulir dibuat bertahap agar meminimalkan kesalahan input dan memastikan hanya satu tim aktif untuk setiap tahun anggaran.



Gambar 3.50. Form tambah penetapan tim penilai

## 3. Halaman admin penetapan kode akun signifikan

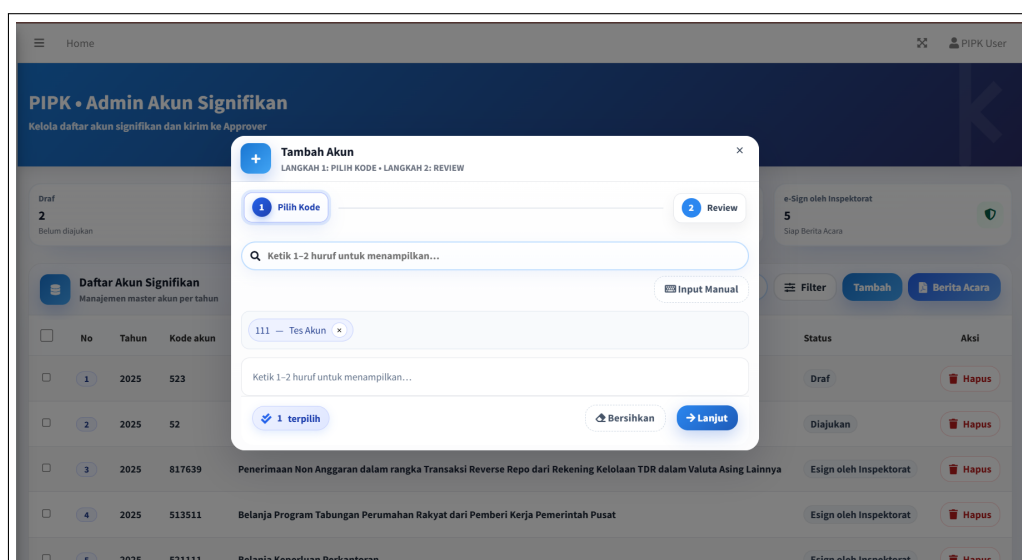
Gambar 3.51 menampilkan halaman admin penilai untuk mengelola daftar kode akun signifikan. Admin dapat menambahkan akun dari basis data atau melalui input manual. Setiap akun memiliki status proses, seperti draf, diajukan, atau telah ditandatangani secara elektronik.



Gambar 3.51. Halaman penetapan kode akun signifikan

#### 4. Form tambah kode akun signifikan

Pada Gambar 3.52, admin penilai dapat memilih kode akun dari database atau menambahkannya secara manual. Akun yang dipilih dapat disimpan sebagai draf untuk ditinjau kembali atau langsung diajukan ke ketua tim penilai.

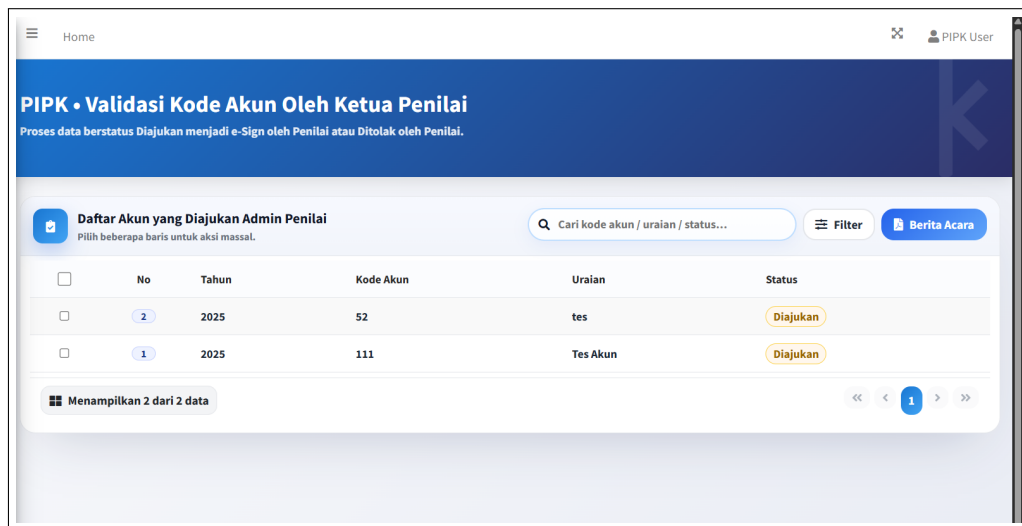


Gambar 3.52. Form tambah kode akun signifikan

#### 5. Proses *review* kode akun oleh ketua tim penilai

Gambar 3.53 menunjukkan halaman ketua tim penilai dalam melakukan validasi

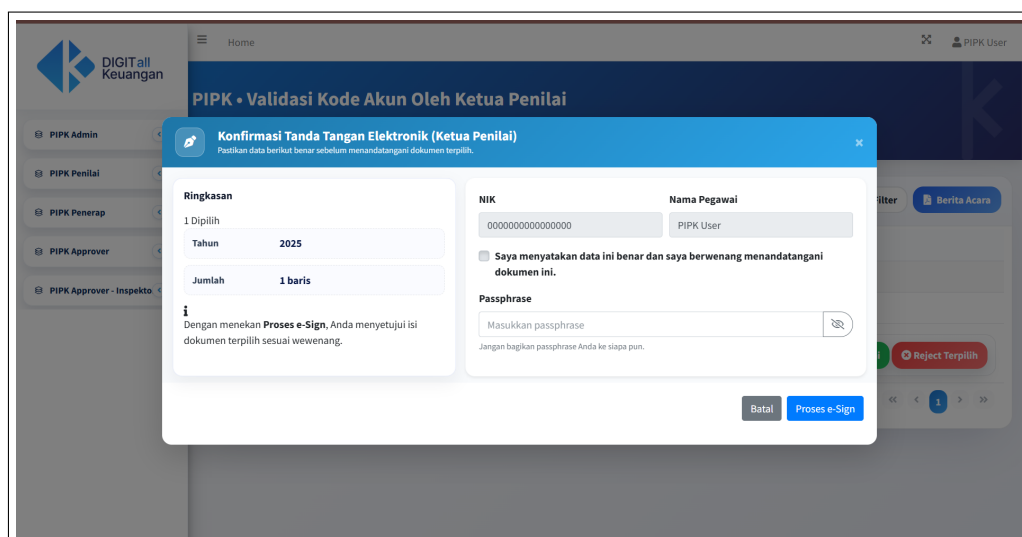
kode akun. Ketua tim dapat menyetujui seluruh akun atau menolak sebagian maupun seluruhnya. Penolakan akan mengembalikan data ke admin penilai untuk diperbaiki.



Gambar 3.53. Validasi kode akun oleh ketua tim penilai

#### 6. Konfirmasi tanda tangan elektronik ketua tim penilai

Gambar 3.54 menampilkan dialog konfirmasi tanda tangan elektronik. Ketua tim penilai diwajibkan melakukan verifikasi identitas dan memasukkan passphrase sebelum menyelesaikan proses e-Sign.

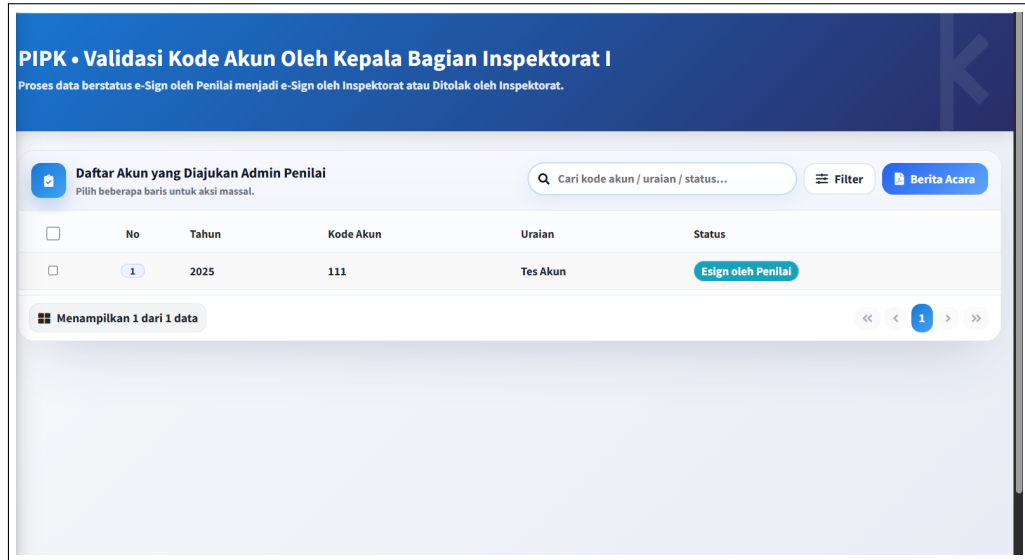


Gambar 3.54. Konfirmasi e-Sign oleh ketua tim penilai

#### 7. Validasi kode akun oleh inspektoral

Gambar 3.55 menunjukkan halaman validasi oleh inspektoral. Inspektoral berperan

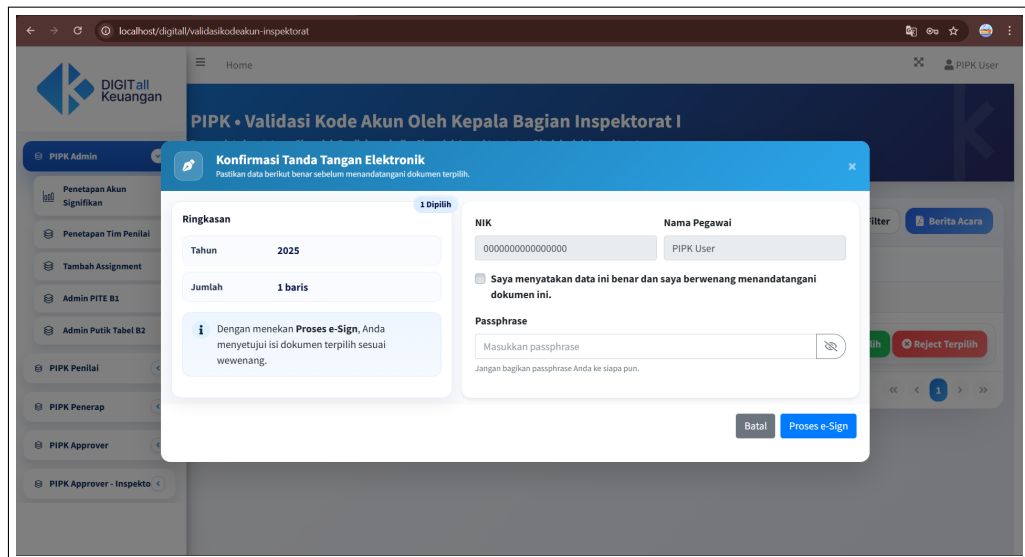
sebagai pihak final yang menentukan apakah kode akun dapat disahkan atau dikembalikan untuk perbaikan.



Gambar 3.55. Validasi kode akun oleh inspektorat

#### 8. Konfirmasi *e-Sign* inspektorat

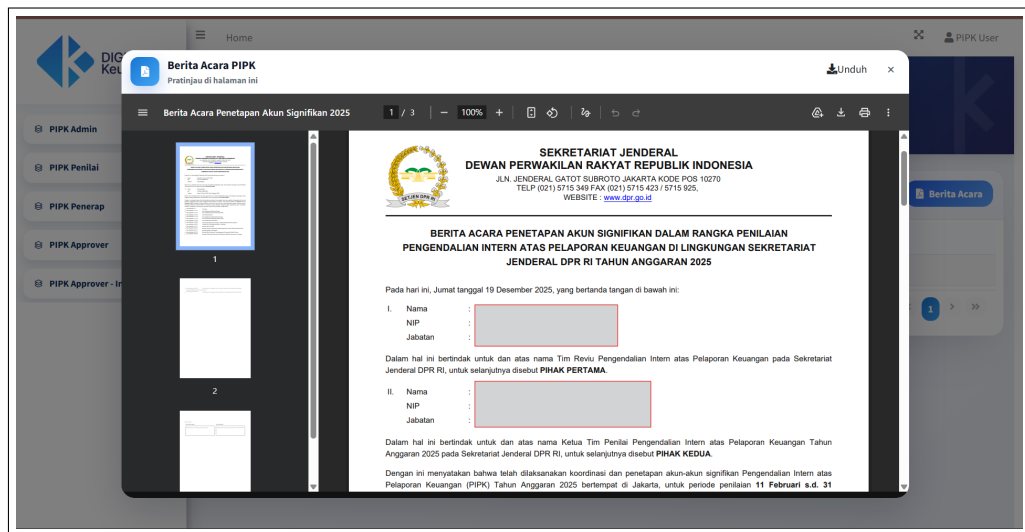
Pada Gambar 3.56, inspektorat melakukan tanda tangan elektronik. Setelah tahap ini selesai, akun signifikan dinyatakan sah dan siap digunakan dalam proses *assignment*.



Gambar 3.56. Konfirmasi e-Sign oleh inspektorat

#### 9. Dokumen berita acara penetapan akun signifikan

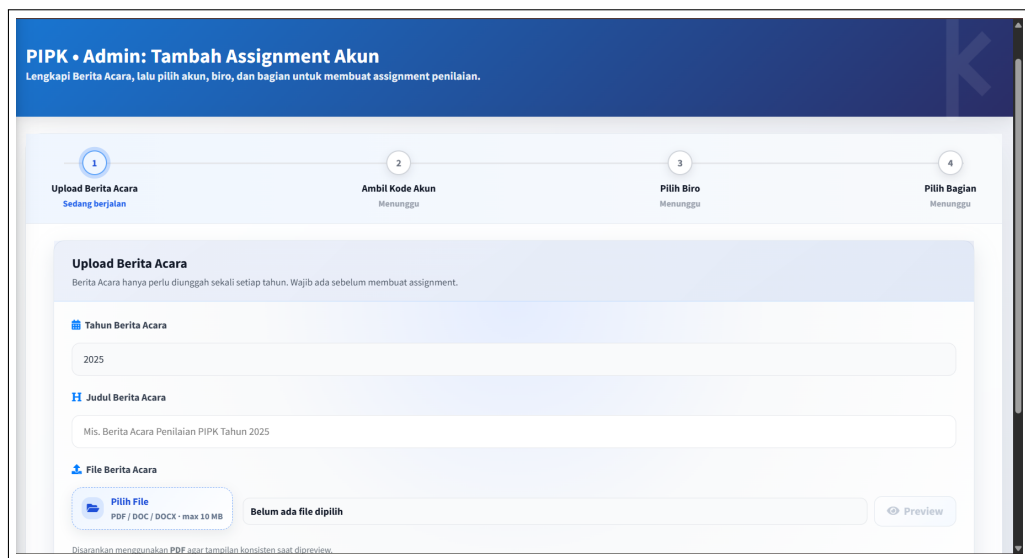
Gambar 3.57 menampilkan dokumen berita acara yang dihasilkan secara otomatis. Dokumen ini menjadi dasar legal untuk melakukan penugasan akun ke unit kerja.



Gambar 3.57. Berita acara penetapan akun signifikan

#### 10. Halaman tambah *assignment* akun

Gambar 3.58 menunjukkan langkah pertama pembuatan *assignment*, yaitu pengunggahan berita acara. Sistem memastikan bahwa berita acara telah tersedia sebelum melanjutkan ke tahap berikutnya.



Gambar 3.58. Langkah upload berita acara

#### 11. Pemilihan kode akun untuk *assignment*

Pada Gambar 3.59, admin hanya dapat memilih kode akun yang telah ditandatangani oleh inspektorat. Pembatasan ini memastikan bahwa *assignment* hanya dibuat berdasarkan akun yang sah.



Gambar 3.59. Pemilihan kode akun *assignment*

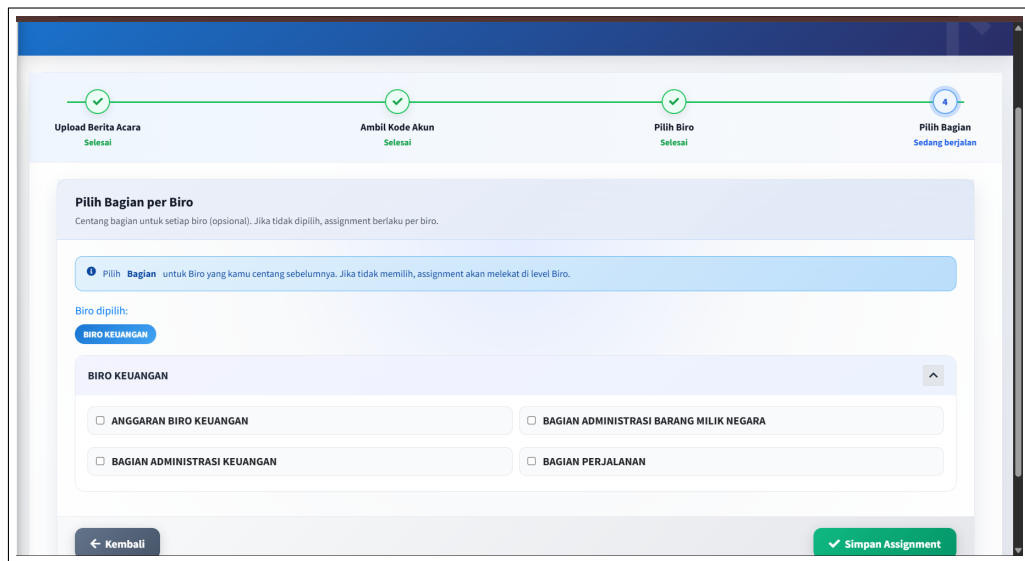
## 12. Pemilihan biro

Gambar 3.60 menunjukkan tahap pemilihan biro sebagai unit kerja penerima *assignment*. Admin dapat memilih satu atau beberapa biro sesuai kebutuhan penilaian.

Gambar 3.60. Pemilihan biro untuk *assignment*

## 13. Pemilihan bagian per biro

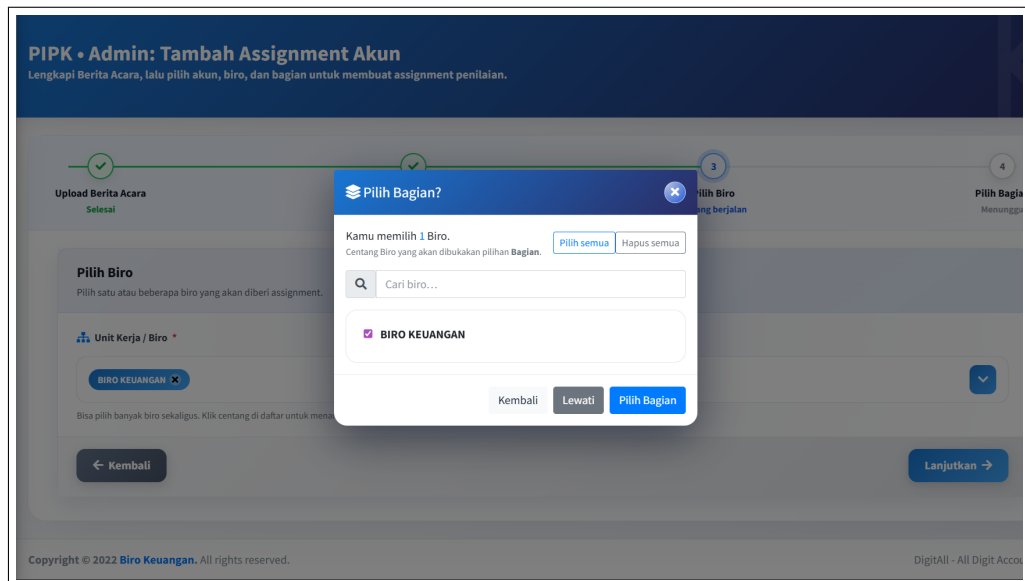
Pada Gambar 3.61, admin dapat menentukan apakah *assignment* berlaku untuk seluruh bagian dalam satu biro atau hanya bagian tertentu. Mekanisme ini memberikan fleksibilitas distribusi tugas.



Gambar 3.61. Pemilihan bagian per biro

#### 14. *Popup* konfirmasi pemilihan bagian

Gambar 3.62 menampilkan dialog konfirmasi sebelum admin melanjutkan proses *assignment*. Dialog ini mencegah kesalahan pemetaan akun ke unit kerja.



Gambar 3.62. *Popup* konfirmasi pemilihan bagian

#### 15. Halaman penerap daftar akun ditugaskan

Gambar 3.63 menampilkan daftar akun yang harus diisi oleh penerap. Setiap baris menunjukkan status pengisian matriks dan akses menuju formulir Tabel A atau Tabel A1.

**e-PIPK Penerap**  
Isi atau tinjau matriks pengendalian untuk tiap akun signifikan.

Tampilkan Tahun Pencarian  
10 2025 Cari satker, unit kerja, kode akun, atau uraian...

**Daftar Akun Ditugaskan**  
Penugasan & status pengisian matriks oleh Penerap Total: 16 data

NO	TAHUN	SATKER	UNIT KERJA	KODE AKUN	URAIAN	STATUS	AKSI
16	2025	Dewan	BIRO HUKUM DAN PENGADUAN MASYARAKAT	522142	Belanja Fasilitas Sewa Rumah untuk Perwakilan Republik Indonesia di Luar Negeri	Beberapa pengajuan ditolak Approver (Tabel A1)	<a href="#">Isi A1</a> <a href="#">Log</a>
15	2025	Setjen	BIRO KEUANGAN Bagian: BAGIAN PERJALANAN	111	Tes Akun	Dalam Proses Pengisian	<a href="#">Isi A</a> <a href="#">Log</a>
14	2025	Setjen	BIRO KESEKRETARIATAN PIMPINAN	111111	Kas di Rekening KUN dalam Rupiah	Dalam Proses Pengisian	<a href="#">Isi A</a> <a href="#">Log</a>
13	2025	Setjen	BIRO KEUANGAN	817639	Penerimaan Non Anggaran dalam rangka Transaksi Reverse Repo dari Rekening Kelolaan TDR dalam Valuta Asing Lainnya	Sudah di Esign oleh Approver dan Penilai	<a href="#">Lihat</a> <a href="#">Log</a>
12	2025	Setjen	BIRO KEUANGAN	827912	Pengeluaran Non Anggaran Pihak Ketiga karena Keterbatasan Sistem Perbankan	Sudah di Esign oleh Approver dan Penilai	<a href="#">Lihat</a> <a href="#">Log</a>

Gambar 3.63. Daftar akun ditugaskan kepada penerap

#### 16. Form pengisian matriks Tabel A

Gambar 3.64 menunjukkan antarmuka pengisian matriks risiko dan pengendalian. Penerap wajib mengisi setiap kolom sesuai ketentuan sebelum melakukan pengajuan.

**Matriks Akun Signifikan**  
Identifikasi Risiko dan Kecukupan Rancangan Pengendaliannya

[Kembali](#) [Table A](#) [Table A1](#)

Satker: Dewan Biro: BIRO HUKUM DAN PENGADUAN MASYARAKAT Akun: 522142 — Belanja Fasilitas Sewa Rumah untuk Perwakilan Republik Indonesia di Luar Negeri

NO	PROSES / TRANSAKSI UTAMA	RISIKO UTAMA	NAMA PENGENDALIAN UTAMA	APLIKASI PENDUKUNG	PELAKSANA PENGENDALIAN	DOKUMEN PENDUKUNG	TIPE PENGENDALIAN	ASERSI
1	tes	tes	tes	Sitangguh	tes	- berkas sop	Manual	- Ketengkapar

Tulis per baris dan wajib diawali tanda minus - Nama Dokumen: [Bisa pilih lebih >](#)

[+ Tambah Baris](#) [Submit](#)

Gambar 3.64. Form pengisian matriks Tabel A

#### 17. Halaman approver daftar pengajuan

Gambar 3.65 menampilkan daftar akun yang telah diajukan oleh penerap. Approver dapat memantau status pengisian dan menentukan akun yang siap diverifikasi.

No	Tahun	Satker	Unit Kerja	Kode Akun	Uraian	Status	Ronde	Aksi
16	2025	Dewan	BIRO HUKUM DAN PENGADUAN MASYARAKAT	522142	Belanja Fasilitas Sewa Rumah untuk Perwakilan Rep...	Belanja pengajuan ditolak Approver (Tabel A1)	A1	Verifikasi, e-Sign, Log
15	2025	Setjen	BIRO KEUANGAN	111	Tes Akun	Dalam Proses Pengajuan	A	Verifikasi, e-Sign, Log
14	2025	Setjen	BIRO KESEKRETARIATAN PIMPINAN	111111	Kas di Rekening KUN dalam Rupiah	Dalam Proses Pengajuan	A	Verifikasi, e-Sign, Log

Gambar 3.65. Daftar pengajuan akun oleh approver

#### 18. Halaman verifikasi pengajuan

Pada Gambar 3.66, approver melakukan pemeriksaan detail matriks. Jika ditemukan data tidak memadai, approver dapat mengembalikan baris tertentu menjadi Tabel A1.

UTAMA	NAMA PENGENDALIAN UTAMA	APLIKASI PENDUKUNG	PELAKSANA PENGENDALIAN	DOKUMEN PENDUKUNG	TIPE PENGENDALIAN	ASERSI	MEMADAI	ALASAN (WAJIB JIKA TIDAK)
	tes	Sitangguh	tes	- berkas ssp	Manual	Kelengkapan (Completeness)	Ya Tidak	

Gambar 3.66. Verifikasi matriks oleh approver

Mekanisme revisi Tabel A1 Ketika sebagian data ditolak, sistem secara otomatis mengarahkan penerap ke Tabel A1. Mekanisme ini memungkinkan perbaikan dilakukan secara terfokus tanpa mengulang seluruh matriks. Alur iteratif pengisian dan verifikasi. Jika hasil revisi masih belum sesuai, proses akan berulang antara penerap dan approver hingga seluruh data dinyatakan memadai. Setelah itu, data siap untuk dilanjutkan ke tahap berikutnya.

### G Implementasi *Backend* Modul PIPK (Tim Penilai, Penetapan Akun, *Assignment*, Penerap, dan Approver)

Bagian ini menjelaskan alur logika pada modul PIPK mulai dari penetapan tim penilai, penetapan kode akun signifikan hingga terbentuknya *assignment* untuk penerap, dan proses verifikasi oleh approver. Setiap langkah ditulis ringkas dan disertai pseudocode agar alur proses mudah ditelusuri.

1. Penetapan tim penilai per tahun (Ketua dan Anggota).

Tahap ini membentuk satu tim penilai pada tahun berjalan menggunakan tabel `pipk_tim_penilai` dan `pipk_tim_penilai_anggota`. Data ketua diperlukan sebagai dasar pemberian kewenangan *e-sign* pada proses penetapan kode akun signifikan lihat Pseudocode 3.12.

```
1 FUNCTION simpanTimPenilai(request):
2     tahun      = request.tahunanggaran
3     ketuaId    = request.ketua_user_id
4     anggota    = request.anggota_user_ids (list)
5
6     VALIDASI:
7         - tahun wajib
8         - ketua wajib
9         - anggota minimal 1
10
11     TRANSAKSI:
12         tim = UPSERT pipk_tim_penilai
13             SET tahunanggaran = tahun,
14                 ketua_user_id = ketuaId
15
16         HAPUS semua pipk_tim_penilai_anggota WHERE
17         tim_id = tim.id
18         UNTUK setiap userId di anggota:
19             INSERT pipk_tim_penilai_anggota (tim_id,
20             user_id)
```

Kode 3.12: Pseudocode simpan tim penilai

Contoh: jika tahun 2025 belum memiliki tim, sistem membuat satu record baru dan menyimpan daftar anggota. Jika tim sudah ada, ketua dan anggota diperbarui tanpa membuat duplikasi.

2. Penetapan kode akun signifikan oleh Admin Penilai (draft/ajukan).

Admin Penilai menambahkan kode akun melalui dua cara: memilih dari basis data (akusignifikan) atau input manual. Hasilnya disimpan dalam tabel `pipk_akun_admin` dengan kolom status untuk membedakan *draft* dan diajukan. Tahap ini menjadi sumber utama sampai *e-sign* oleh Inspektorat lihat Pseudocode 3.13.

```
1 FUNCTION simpanAkunAdmin(request):
2     tahun      = request.tahunanggaran
3     mode       = request.mode // 'db' atau 'manual'
4     aksi       = request.aksi  // 'draft' atau 'ajukan'
5
```

```

6      IF mode == 'db':
7          akun = SELECT akun, uraian FROM akunsignifikan
            WHERE akun = request.kode_akun
8      ELSE:
9          akun.akun      = request.kode_akun_manual
10         akun.uraian     = request.uraian_manual
11
12     VALIDASI:
13         - tahun wajib
14         - akun.akun wajib
15         - akun tidak boleh kosong
16
17     status = IF aksi == 'draft' THEN 'DRAFT' ELSE '
            DIAJUKAN'
18
19     UPSERT pipk_akun_admin
20         SET tahunanggaran = tahun,
21             akun            = akun.akun,
22             uraian_akun     = akun.uraian,
23             status          = status,
24             updated_at      = now()
25
26     RETURN sukses

```

### Kode 3.13: Pseudocode tambah akun signifikan (draft/ajukan)

Contoh: saat pengguna memilih akun dari akunsignifikan lalu klik Simpan Draft, status menjadi *draft*. Jika klik Simpan & Ajukan, status menjadi diajukan untuk menunggu *e-sign* ketua tim penilai.

#### 3. *E-sign* Ketua Tim Penilai (setujui/reject sebagian/seluruh).

Ketua tim penilai melakukan validasi atas data yang diajukan. Sistem menerima daftar baris terpilih, lalu mengubah status pada pipk\_akun\_admin. Jika ada yang ditolak, status dikembalikan agar Admin Penilai melakukan perbaikan; jika disetujui maka lanjut ke tahap Inspektorat lihat Pseudocode 3.14.

```

1 FUNCTION esignKetuaTim(request):
2     tahun      = request.tahunanggaran
3     daftarAkun = request.kode_akun_list
4     keputusan  = request.keputusan // 'setujui' atau '
            reject'
5     passphrase = request.passphrase
6
7     VALIDASI:
8         - user login adalah ketua tim pada tahun
            tersebut

```

```

9      - passphrase valid (sesuai mekanisme e-Sign)
10     - daftarAkun tidak kosong
11
12     UNTUK setiap kodeAkun di daftarAkun:
13         IF keputusan == 'setujui':
14             UPDATE pipk_akun_admin
15             SET status = 'ESIGN_KETUA',
16                 esign_ketua_by = user.id,
17                 esign_ketua_at = now()
18             WHERE tahunanggaran = tahun AND akun =
kodeAkun
19         ELSE:
20             UPDATE pipk_akun_admin
21             SET status = 'REJECT_KETUA',
22                 catatan_reject = request.catatan (
opsional),
23                 updated_at = now()
24             WHERE tahunanggaran = tahun AND akun =
kodeAkun
25
26     RETURN sukses

```

Kode 3.14: Pseudocode e-Sign ketua tim penilai

Contoh: Ketua tim menolak 2 akun dari 10 akun yang diajukan, maka 8 akun berstatus "e-sign oleh ketua tim penilai" dan 2 akun berstatus "ditolak oleh ketua tim penilai" untuk diperbaiki Admin.

#### 4. *E-sign* Inspektorat I (validasi akhir kode akun).

Inspektorat melakukan verifikasi lanjutan. Jika menyetujui, status menjadi final sebagai akun yang sah dipakai pada berita acara; jika menolak, status dikembalikan ke Admin untuk koreksi (melalui status *reject*). Setelah semua disetujui, berita acara dapat dihasilkan lihat Pseudocode 3.15.

```

1 FUNCTION esignInspektorat(request):
2     tahun      = request.tahunanggaran
3     daftarAkun = request.kode_akun_list
4     keputusan  = request.keputusan // 'setujui' atau '
reject'
5     passphrase = request.passphrase
6
7     VALIDASI:
8         - user berperan inspektorat pada modul ini
9         - passphrase valid
10        - akun yang diproses minimal status '
ESIGN_KETUA'

```

```

11
12     UNTUK setiap kodeAkun di daftarAkun:
13         IF keputusan == 'setujui':
14             UPDATE pipk_akun_admin
15             SET status = 'ESIGN_INSPEKTORAT',
16                 esign_inspektorat_by = user.id,
17                 esign_inspektorat_at = now()
18             WHERE tahunanggaran = tahun AND akun =
kodeAkun
19         ELSE:
20             UPDATE pipk_akun_admin
21             SET status = 'REJECT_INSPEKTORAT',
22                 catatan_reject = request.catatan (
opsional),
23                 updated_at = now()
24             WHERE tahunanggaran = tahun AND akun =
kodeAkun
25
26     RETURN sukses

```

Kode 3.15: Pseudocode e-Sign inspektorat

Contoh: jika seluruh akun sudah *e-sign* oleh inspektorat, maka sistem menganggap daftar akun signifikan tahun tersebut resmi dan dapat dimasukkan ke berita acara.

##### 5. Pembuatan dan penyimpanan Berita Acara penetapan akun.

Berita acara disimpan pada tabel `pipk_berita_acara_penilai`. Dokumen ini digunakan sebagai prasyarat pada pembuatan *assignment* akun lihat Pseudocode 3.16.

```

1 FUNCTION simpanBeritaAcara(request):
2     tahun    = request.tahunanggaran
3     judul    = request.judul
4     filePdf  = request.file
5
6     VALIDASI:
7         - tahun wajib
8         - file wajib (pdf/doc/docx), ukuran <= batas
9         - berita acara per tahun hanya satu (aturan
sistem)
10
11     path = simpanFileKeStorage(filePdf)
12
13     UPSERT pipk_berita_acara_penilai
14         SET tahunanggaran = tahun,
15             judul          = judul,
16             file_path      = path,

```



```

17         updated_at      = now ()
18
19     RETURN sukses + url preview

```

### Kode 3.16: Pseudocode simpan berita acara

Contoh: tahun 2025 diunggah sekali, jika diunggah ulang maka record diperbarui agar versi terbaru digunakan pada assignment.

#### 6. Step tambah assignment.

Setelah berita acara tersedia, admin membuat *assignment* agar akun masuk ke daftar kerja penerap. Kode akun yang dapat dipilih dibatasi hanya akun yang sudah disetujui oleh inspektorat. Pemilihan biro mengambil dari tabel biro, dan bagian dari tabel bagian. Hasil akhirnya disimpan pada pipk\_akun\_signifikan. Pemetaan pembatasan penerap per bagian memanfaatkan pipk\_akun\_bagian lihat Pseudocode 3.17.

```

1 FUNCTION simpanAssignment(request):
2     tahun      = request.tahunanggaran
3     kodeAkun   = request.kode_akun
4     biroIds    = request.idbiro_list
5     bagianMap  = request.bagian_per_biro // map: idbiro
6             -> list idbagian (opsional)
7
8     VALIDASI:
9         - berita acara tahun tersebut sudah ada
10        - kodeAkun status di pipk_akun_admin = '
11        ESIGN_INSPEKTORAT'
12        - biroIds tidak kosong
13
14    TRANSAKSI:
15        UNTUK setiap idbiro di biroIds:
16            INSERT pipk_akun_signifikan(
17                tahunanggaran, kode_akun, idbiro,
18                status_table
19            ) VALUES (tahun, kodeAkun, idbiro, 'A')
20
21            // opsional: jika user memilih bagian
22            tertentu untuk biro tsb
23            IF bagianMap[idbiro] tidak kosong:
24                UNTUK setiap idbagian di bagianMap[
25                idbiro]:
26                    INSERT pipk_akun_bagian(
27                        tahunanggaran, kode_akun,
28                        idbiro, idbagian
29                    )

```

```

24
25 RETURN sukses

```

### Kode 3.17: Pseudocode simpan assignment akun signifikan

Contoh: jika admin memilih Biro Keuangan dan memilih bagian Bagian Perjalanan, maka akun akan muncul hanya pada penerap dengan idbiro=Keuangan dan idbagian=Perjalanan. Jika bagian tidak dipilih, akun berlaku untuk seluruh bagian dalam biro tersebut (sesuai aturan tampilan penerap).

#### 7. Penerap mengisi matriks (Tabel A) dan submit.

Penerap mengisi matriks pengendalian dan menyimpan ke pipk\_table\_a. Status *assignment* tetap berada pada siklus A/A1 sampai dinyatakan memadai oleh approver. Proses simpan baris memakai pola *upsert per baris* agar pembaruan tidak menambah duplikasi lihat Pseudocode 3.18.

```

1 FUNCTION simpanMatriksA(request):
2     tahun      = request.tahunanggaran
3     kodeAkun   = request.kode_akun
4     rows       = request.rows // list baris matriks
5
6     VALIDASI:
7         - user penerap berhak akses (cek idbiro/
          idbagian)
8         - rows minimal 1
9
10    TRANSAKSI:
11        UNTUK setiap r di rows:
12            UPSERT pipk_table_a BY (tahun, kodeAkun,
          nomor_baris)
13                SET proses_transaksi = r.proses,
14                    risiko_utama      = r.risiko,
15                    pengendalian      = r.pengendalian,
16                    aplikasi          = r.aplikasi,
17                    pelaksana         = r.pelaksana,
18                    dokumen           = r.dokumen,
19                    tipe_pengendalian= r.tipe,
20                    asersi            = r.asersi,
21                    updated_at        = now()
22
23        UPDATE pipk_akun_signifikan
24        SET status_table = 'A', updated_at = now()
25        WHERE tahunanggaran = tahun AND kode_akun =
          kodeAkun
26        AND sesuai scope biro/bagian
27

```

### Kode 3.18: Pseudocode simpan matriks penerap (Tabel A)

Contoh: penerap menambah baris baru lalu klik submit, sistem menyimpan ke pipk\_table\_a dan siap diverifikasi approver.

8. Approver memverifikasi matriks: memadai atau revisi ke Tabel A1 (loop).  
Approver mengecek kelengkapan. Jika ada baris tidak memadai, approver mengisi catatan pada kolom catatan\_approver dan mengubah memadai\_approver=0. Sistem menyimpan detail revisi ke pipk\_table\_a1 agar penerap dapat fokus memperbaiki baris tertentu. Siklus ini dapat berulang (loop) sampai seluruh baris dinyatakan memadai lihat Pseudocode 3.19.

```

1 FUNCTION verifikasiApprover(request):
2     tahun      = request.tahunanggaran
3     kodeAkun   = request.kode_akun
4     keputusan  = request.keputusan // 'setujui' atau '
        tolak_sebagian'
5     catatanPerBaris = request.catatan_map //
        nomor_baris -> catatan
6
7     VALIDASI:
8         - user approver berhak atas biro/bagian terkait
9
10    TRANSAKSI:
11        IF keputusan == 'setujui':
12            UPDATE semua pipk_table_a
13            SET memadai_approver = 1,
14                catatan_approver = NULL
15            WHERE tahunanggaran = tahun AND kode_akun =
        kodeAkun
16
17            UPDATE pipk_akun_signifikan
18            SET status_table = 'A', updated_at = now()
19            WHERE tahunanggaran = tahun AND kode_akun =
        kodeAkun
20
21        ELSE:
22            // tandai baris yang tidak memadai
23            UNTUK setiap (noBaris, catatan) di
        catatanPerBaris:
24                UPDATE pipk_table_a
25                SET memadai_approver = 0,
26                    catatan_approver = catatan
27                WHERE tahunanggaran = tahun AND

```

```

kode_akun = kodeAkun
28         AND nomor_baris = noBaris
29
30         UPSERT pipk_table_a1 BY (tahun,
kodeAkun, nomor_baris)
31         SET catatan_revisi = catatan,
32             status_revisi = '
PERLU_PERBAIKAN',
33             updated_at = now()
34
35         UPDATE pipk_akun_signifikan
36         SET status_table = 'A1', updated_at = now()
37         WHERE tahunanggaran = tahun AND kode_akun =
kodeAkun
38
39     RETURN sukses

```

Kode 3.19: Pseudocode verifikasi approver (A/A1)

Contoh: jika 3 baris tidak memadai, maka status menjadi A1. Penerap memperbaiki baris terkait lalu submit ulang, dan approver memverifikasi kembali hingga seluruh baris memadai.

### 3.4 Kendala dan Solusi yang Ditemukan

Selama proses pengembangan sistem DIGITall, ditemukan beberapa kendala teknis dan non-teknis yang memengaruhi jalannya pekerjaan. Kendala-kendala tersebut muncul pada tahap integrasi antar modul, sinkronisasi data pengguna, serta pengujian sistem. Rincian kendala yang dihadapi dijelaskan sebagai berikut.

1. Penyesuaian fitur antar modul membutuhkan waktu lebih lama karena setiap modul (Berita, Lembur, dan PIPK) memiliki struktur, alur kerja, dan logika yang berbeda. Proses integrasi juga harus menyesuaikan dengan format data serta hak akses masing-masing peran pengguna.
2. Sinkronisasi data antar pengguna sempat menimbulkan inkonsistensi tampilan dan duplikasi data, terutama saat beberapa pengguna mengakses atau memperbarui data secara bersamaan tanpa validasi otomatis.
3. Pada tahap pengujian awal, terjadi kendala teknis berupa kesalahan validasi *form* dan koneksi basis data. Hal ini menyebabkan beberapa fitur tidak dapat dijalankan dengan stabil di lingkungan pengujian.

Berdasarkan kendala tersebut, langkah-langkah penyelesaian yang dilakukan adalah sebagai berikut.

1. Dilakukan pembagian tugas secara terstruktur dan dokumentasi alur kerja tiap modul agar pengembangan lebih terarah dan mudah disinkronkan antar tim pengembang.

2. Ditambahkan mekanisme validasi otomatis dan pembaruan data secara *real-time* untuk mencegah inkonsistensi serta memastikan integritas data antar pengguna.
3. Diperbaiki konfigurasi koneksi basis data dan dilakukan *debugging* bertahap melalui uji coba internal hingga seluruh fitur berjalan stabil sesuai rancangan sistem.