**LAPORAN PROYEK CAPSTONE**

**SEMESTER GENAP TAHUN AKADEMIK 2024/2025**

**JUDUL PROYEK**

**(SISTEM FILE ARCHIVE PERUSAHAAN)**

****

**Disusun oleh:**

Abi Albani (23416255201068)

Muhammad Ammar Fauzan (23416255201047)

Giovany Kahlil Gibran (23416255201046)

Junico Hotasi Sitohang (23416255201251)

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS BUANA PERJUANGAN KARAWANG**

**2025**

# KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya sehingga laporan ini dapat diselesaikan dengan baik. Laporan ini disusun sebagai bagian dari dokumentasi hasil kegiatan [sebutkan topik laporan, misalnya: "pengembangan sistem File Archive Perusahaan "], yang bertujuan untuk memberikan gambaran mengenai proses kerja, metode yang digunakan, dan hasil yang telah dicapai selama pelaksanaan kegiatan

Tujuan dari penyusunan laporan ini adalah untuk memenuhi kewajiban akademik serta sebagai bentuk evaluasi terhadap kegiatan yang telah dilakukan. Harapannya, laporan ini dapat memberikan manfaat bagi penulis dan pembaca dalam mengembangkan pemahaman tentang topik yang dibahas.

Kami menyadari bahwa keberhasilan penyusunan laporan ini tidak terlepas dari dukungan berbagai pihak. Oleh karena itu, dengan tulus kami menyampaikan ucapan terima kasih kepada:

1. **Dosen pembimbing**, yang telah memberikan bimbingan, arahan, serta saran yang sangat berharga selama proses pengerjaan kegiatan hingga penyusunan laporan ini.

Kami juga menyadari bahwa laporan ini masih memiliki kekurangan. Oleh karena itu, kami terbuka terhadap kritik dan saran yang dapat membantu menyempurnakan karya ini di masa depan.

Semoga laporan ini dapat memberikan manfaat dan menjadi referensi yang berguna bagi semua pihak.

Karawang, [2025]

Kelompok 43

# DAFTAR ISI

[KATA PENGANTAR 2](#_Toc201961372)

[DAFTAR ISI 3](#_Toc201961373)

[BAB I PENDAHULUAN 4](#_Toc201961374)

[1.1 Latar Belakang 4](#_Toc201961375)

[1.2 Rumusan Masalah 5](#_Toc201961376)

[1.3 Tujuan 5](#_Toc201961377)

[1.4 Manfaat 5](#_Toc201961378)

[1.5 Stakeholder 6](#_Toc201961379)

[1.6 Ringkasan Kebutuhan Pengguna 6](#_Toc201961380)

[BAB II PERENCANAAN PROYEK 8](#_Toc201961381)

[2.1 Jadwal Proyek 8](#_Toc201961382)

[2.2 Pembagian Tugas Tim 8](#_Toc201961383)

[2.3 Perencanaan Sumber Daya 9](#_Toc201961384)

[2.4 Risiko dan Antisipasi 10](#_Toc201961385)

[BAB III ANALISIS DAN PERANCANGAN SISTEM 12](#_Toc201961386)

[3.1 Use Case Diagram dan Deskripsinya 12](#_Toc201961387)

[3.2 Activity Diagram 18](#_Toc201961388)

[3.3 Class Diagram 22](#_Toc201961389)

[3.4 Mockup UI/UX 25](#_Toc201961390)

[3.4.1 Mockup UI/UX Versi Mobile 25](#_Toc201961391)

[3.4.2 Mockup UI/UX Tampilan Website 27](#_Toc201961392)

[3.5 Rancangan Basis Data 33](#_Toc201961393)

[3.5.1 Penjelasan Struktur Database 33](#_Toc201961394)

[3.5.2 Tabel-Tabel Utama dan Atributnya 34](#_Toc201961395)

[3.5.3 ERD (Entity Relationship Diagram) 38](#_Toc201961396)

[3.5.4 Skrip Sql 38](#_Toc201961397)

[3.6 Perancangan API 41](#_Toc201961398)

[3.6.1 Struktur Route di Laravel (routes/api.php) 41](#_Toc201961399)

[3.6.2 Dokumentasi API Laravel (Endpoint, Method, Parameter, Response) 43](#_Toc201961400)

[BAB IV IMPLEMENTASI SISTEM 44](#_Toc201961401)

[4.1 Teknologi yang Digunakan 44](#_Toc201961402)

[4.2 Implementasi Fitur-Fitur Utama 45](#_Toc201961403)

[a. Fitur Login 45](#_Toc201961404)

[b. Fitur CRUD Template Surat 46](#_Toc201961405)

[c. Fitur Pengajuan Surat 46](#_Toc201961406)

[d. Fitur Validasi Surat oleh Admin 46](#_Toc201961407)

[e. Fitur Riwayat dan Status Surat 46](#_Toc201961408)

[4.3 Implementasi Backend dan API 46](#_Toc201961409)

[4.4 Implementasi Aplikasi Mobile 48](#_Toc201961410)

[4.5 Scope of Work Aplikasi 50](#_Toc201961411)

[4.6 Cuplikan Kode Program 50](#_Toc201961412)

[4.6.1 Frontend (Angular / Ionic) 50](#_Toc201961413)

[4.6.2 Backend (Laravel) 58](#_Toc201961414)

[BAB V PENGUJIAN SISTEM 62](#_Toc201961415)

[5.1 Metode Pengujian (Blackbox) 62](#_Toc201961416)

[5.2 Tabel Hasil Pengujian Fitur 62](#_Toc201961417)

[BAB VI PENERAPAN DAN UJI COBA 63](#_Toc201961418)

[6.1 Uji Coba ke Pengguna / Stakeholder 63](#_Toc201961419)

[6.2 Dokumentasi Instalasi / Manual Pengguna 63](#_Toc201961420)

[6.3 Evaluasi Hasil Uji Coba (Feedback Pengguna) 64](#_Toc201961421)

[6.4 Rencana Pengembangan Lebih Lanjut 64](#_Toc201961422)

[BAB VII PENUTUP 65](#_Toc201961423)

[7.1 Kesimpulan 65](#_Toc201961424)

[7.2 Saran 65](#_Toc201961425)

[LAMPIRAN 66](#_Toc201961426)

# BAB I PENDAHULUAN

* 1. Latar Belakang

Dalam aktivitas operasional sebuah perusahaan, kebutuhan terhadap dokumen resmi seperti surat keterangan karyawan, permohonan cuti, surat pengajuan keluhan, hingga surat rekomendasi menjadi bagian penting dari sistem administrasi yang tidak bisa diabaikan. Sayangnya, di banyak perusahaan, pengelolaan dokumen-dokumen ini masih dilakukan secara manual atau belum terdigitalisasi dengan baik, sehingga menimbulkan berbagai permasalahan seperti duplikasi data, ketidakteraturan penomoran surat, kesalahan isi, keterlambatan distribusi, dan kesulitan dalam pelacakan Riwayat arsip.

Permasalahan tersebut diperparah dengan tidak adanya sistem baku untuk pengajuan dan pencatatan surat, sehingga membuat bagian personalia (HRD) atau General Affair (GA) harus bekerja ekstra untuk memastikan keteraturan dan keakuratan arsip surat. Di sisi lain, karyawan sebagai pengguna pun sering mengalami kesulitan saat hendak mengakses template surat resmi, karena format yang tersebar tidak standar atau bahkan hilang.

Melihat kondisi tersebut, dibutuhkan sebuah solusi digital yang dapat mengotomatisasi proses administrasi surat menyurat di perusahaan. Solusi ini berupa **Sistem File Archive Perusahaan**, sebuah aplikasi yang dirancang untuk mempermudah proses pengelolaan template surat, pengajuan surat resmi, serta pelacakan riwayat dokumen dengan sistem penomoran otomatis dan terstandarisasi.

Sistem ini dibangun dengan pendekatan teknologi modern yang terdiri dari:

* **Frontend mobile (Ionic + Angular)**, agar karyawan dapat login, melihat, mengunduh, dan mengajukan surat secara langsung melalui perangkat mereka.
* **Backend API (Laravel / Django Rest Framework)**, sebagai pengelola utama data, autentikasi, logika penomoran surat, hingga histori pengajuan.
* **Database (MySQL)**, untuk menyimpan seluruh informasi penting seperti data pengguna, template surat, nomor surat, dan log aktivitas.

Melalui sistem ini, proses administratif dapat dilakukan secara **real-time**, aman, dan terarsipkan secara rapi. Dengan adanya fitur **role-based access**, sistem juga mampu membedakan antara hak akses karyawan dan admin HRD/GA secara terstruktur. Selain itu, sistem penomoran otomatis yang menggunakan format seperti HRD/2025/001 menjamin tidak terjadinya nomor surat yang ganda atau tidak berurutan.

Pengembangan aplikasi ini menjadi penting sebagai langkah nyata untuk meningkatkan efisiensi, transparansi, dan akuntabilitas dalam manajemen dokumen perusahaan. Selain memudahkan pihak HRD dalam mengelola dan memverifikasi surat, aplikasi ini juga memberikan kenyamanan bagi karyawan untuk mengakses dokumen dengan cara yang cepat dan sistematis, sehingga menunjang produktivitas kerja di lingkungan perusahaan.

1.2 Rumusan Masalah

1. Bagaimana cara membangun sebuah sistem yang dapat memudahkan karyawan dalam mengakses dan mengunduh template surat resmi perusahaan secara digital?
2. Bagaimana merancang sistem pengajuan surat yang otomatis menghasilkan nomor surat unik dan terurut berdasarkan kategori dan tahun?
3. Bagaimana mengelola riwayat pengajuan surat agar dapat diakses kembali dengan mudah oleh karyawan maupun pihak HRD?
4. Bagaimana mengembangkan aplikasi yang membedakan hak akses antara karyawan dan admin (HRD/GA) dengan sistem login berbasis role?
5. Bagaimana memastikan keamanan dan keteraturan arsip surat yang dikelola secara digital dan real-time?

1.3 Tujuan

Tujuan dari pengembangan proyek ini adalah:

1. Membuat aplikasi sistem arsip file perusahaan yang dapat digunakan oleh karyawan untuk melihat, mengunduh, dan mengajukan surat resmi secara online.
2. Menerapkan sistem penomoran surat otomatis yang terstandarisasi untuk setiap jenis dokumen sesuai tahun dan kategori.
3. Menyediakan fitur pelacakan dan riwayat surat bagi pengguna untuk memantau status pengajuan serta mendokumentasikan aktivitas surat menyurat.
4. Menerapkan sistem login berbasis role (karyawan dan admin) guna mengatur hak akses pengguna terhadap fitur-fitur yang tersedia.
5. Mengimplementasikan penyimpanan data berbasis database yang aman, terstruktur, dan mendukung pencarian dokumen secara cepat dan efisien.

1.4 Manfaat

Adapun manfaat dari aplikasi Sistem File Archive Perusahaan ini antara lain:

* **Bagi karyawan**:
  + Mempercepat proses permohonan surat tanpa harus datang langsung ke HRD.
  + Memudahkan pencarian template surat sesuai kebutuhan.
  + Dapat mengetahui status pengajuan secara langsung.
* **Bagi HRD / General Affair**:
  + Menghemat waktu dalam mengelola dan memverifikasi surat pengajuan.
  + Menjaga keteraturan dan keamanan dokumen melalui penomoran otomatis.
  + Dapat menghasilkan laporan pengajuan surat berdasarkan kategori dan waktu.
* **Bagi perusahaan secara umum**:
  + Meningkatkan efisiensi administrasi internal.
  + Mengurangi penggunaan kertas (paperless).
  + Menjamin dokumentasi yang lebih akurat dan terarsip rapi.

1.5 Stakeholder

Berisi daftar pihak-pihak yang terlibat atau berkepentingan terhadap sistem. Sebutkan siapa mereka (misalnya: pengguna, admin, pemilik usaha) dan apa peran atau keterlibatannya.

1.6 Ringkasan Kebutuhan Pengguna

Kebutuhan Pengguna: Karyawan

1. Dapat melakukan login ke dalam aplikasi menggunakan akun yang valid
2. Melihat daftar template surat resmi yang tersedia berdasarkan kategori
3. Mengunduh template surat dalam format PDF atau Word
4. Mengisi formulir pengajuan surat sesuai template yang dipilih
5. Mendapatkan nomor surat resmi otomatis saat mengajukan surat
6. Melihat status pengajuan surat (dalam proses, disetujui, dicetak, dll.)
7. Melihat riwayat surat yang pernah diajukan lengkap dengan nomor & tanggal

Kebutuhan Pengguna: Admin (HRD / General Affair)

1. Dapat melakukan login sebagai admin untuk mengakses fitur manajemen penuh
2. Mengelola template surat: tambah, edit, hapus, dan klasifikasikan
3. Melihat dan memverifikasi pengajuan surat dari karyawan
4. Menyetujui / menolak pengajuan dan memberikan validasi nomor surat otomatis
5. Melakukan cetak surat resmi yang telah disetujui
6. Melihat dan mengelola riwayat seluruh surat masuk dan keluar
7. Membuat laporan pengajuan surat berdasarkan kategori, jumlah, dan tanggal

Kebutuhan Umum Sistem

1. Aplikasi mendukung penggunaan di perangkat mobile (Android/iOS)
2. Sistem memiliki autentikasi berbasis token (JWT) dan login berbasis role
3. Semua arsip surat dan template tersimpan di database atau cloud storage
4. Sistem penomoran surat mengikuti format perusahaan (contoh: HRD/2025/001)
5. Aplikasi mendukung pencarian dan pengurutan surat secara efisien

# BAB II PERENCANAAN PROYEK

2.1 Jadwal Proyek

Berisi perencanaan waktu pelaksanaan proyek dari awal hingga selesai, biasanya dalam bentuk tabel mingguan. Jadwal mencakup tahapan seperti analisis, desain, implementasi, pengujian, dan pelaporan.

|  |  |  |  |
| --- | --- | --- | --- |
| Minggu ke | Kegiatan | Penanggung Jawab | Status |
| 1 | Penyusunan latar belakang dan kebutuhan user | Seluruh tim | Selesai |
| 2 | Analisis sistem lama dan pembuatan use case & activity diagram | Giovany | Selesai |
| 3 | Perancangan class diagram dan struktur basis data | Giovany dan Abi | Selesai |
| 4 | Perancangan UI/UX antarmuka pengguna (mockup halaman login, dashboard, dll) | Abi | Selesai |
| 5 | **Pengerjaan Backend**: Setup database, API login, otentikasi user | Ammar | Selesai |
| 6 | **Pengerjaan Backend**: API surat, template surat, validasi, dan status surat | Ammar | Selesai |
| 7 | **Integrasi Frontend & Backend**, serta pengujian alur end-to-end | Ammar dan Giovany | Selesai |
| 8 | Penyusunan laporan akhir proyek dan dokumentasi teknis | Seluruh Tim | Selesai |
| 9 | Presentasi akhir dan evaluasi proyek | Seluruh Tim | Proses |
| 10 | Lulus Insyaallah | Seluruh Tim | Proses |

2.2 Pembagian Tugas Tim

Agar proyek berjalan dengan efektif dan terarah, setiap anggota tim memiliki peran serta tanggung jawab masing-masing yang disesuaikan dengan keahlian dan kontribusinya. Berikut adalah pembagian kerja selama proyek berlangsung:

1. **Abi Albani**
   * **Tanggung Jawab:**
     + Mendesain **tampilan antarmuka pengguna (UI/UX)** agar aplikasi mudah digunakan dan menarik secara visual.
     + Menyusun dan mengelola **struktur database**, termasuk perancangan skema, pembuatan relasi, dan penyimpanan data yang digunakan oleh sistem.
2. **Giovany Kahlil Gibran**
   * **Tanggung Jawab:**
     + Mengembangkan antarmuka dan fungsionalitas aplikasi menggunakan **framework Ionic**.
     + Menyusun dan menyelesaikan bagian utama dari **laporan akhir proyek**, termasuk dokumentasi teknis terkait pengembangan frontend dan hasil implementasi.
3. **Muhammad Ammar Fauzan**
   * **Tanggung Jawab:**
     + Mengembangkan bagian **backend aplikasi**, termasuk pembuatan API dan logika server untuk mendukung komunikasi antara frontend dan database.
     + Berkontribusi dalam penulisan **laporan akhir proyek**, khususnya bagian yang menjelaskan alur backend dan sistem integrasi secara keseluruhan.

2.3 Perencanaan Sumber Daya

* Berikut adalah daftar **kebutuhan sumber daya** yang digunakan dalam proyek **Sistem File Archive Perusahaan.**

1. Software

* Visual Studio Code – digunakan sebagai text editor utama untuk frontend & backend.
* Laravel – untuk backend development dan server lokal.
* MySQL / phpMyAdmin – untuk pengelolaan database.
* Postman – untuk pengujian API backend.
* Google Chrome – browser untuk menjalankan dan menguji sistem.

2. Hardware

* Laptop / Komputer – sebagai perangkat utama pengembangan.
* Printer (Opsional) – untuk cetak laporan atau hasil surat.
* Koneksi Internet Stabil – untuk kolaborasi tim, update library, dan testing online.

3. Akun Layanan

* GitHub / GitLab – untuk version control dan kolaborasi kode.
* Google Drive – untuk backup file dan dokumentasi proyek.
* Email aktif (Gmail) – untuk testing login, pengiriman notifikasi (opsional).

4. Tools Desain & Dokumentasi

* Figma – untuk desain UI mockup atau presentasi.
* draw.io – untuk membuat use case, class diagram, dan activity diagram.
* Microsoft Word / Google Docs – untuk menulis laporan proyek.

2.4 Risiko dan Antisipasi

Menjelaskan potensi risiko yang mungkin terjadi selama proyek, misalnya keterlambatan, konflik tim, kesalahan implementasi, dan sebagainya. Sertakan solusi atau strategi mitigasi untuk tiap risiko.

* **Keterlambatan Jadwal Proyek**

Risiko paling umum yang terjadi karena waktu pengerjaan yang tidak sesuai rencana.  
**Solusi:** Buat jadwal realistis, lakukan evaluasi mingguan, dan pastikan pembagian tugas jelas sejak awal.

* **Konflik Antar Anggota Tim**

Perbedaan pendapat dan miskomunikasi bisa menghambat koordinasi.  
**Solusi:** Lakukan rapat rutin, jaga komunikasi terbuka, dan tetapkan peran/tanggung jawab masing-masing dengan jelas.

* **Kesalahan Implementasi (Bug atau Error)**

Coding yang tidak tepat bisa menyebabkan fitur tidak berjalan atau terjadi error pada sistem.  
**Solusi:** Terapkan pengujian secara bertahap (unit dan integrasi), gunakan Git untuk version control, dan dokumentasikan proses dengan baik.

* **Kurangnya Kemampuan Teknis Tim**

Jika tim kurang memahami teknologi (backend/frontend), pekerjaan bisa terhambat.  
**Solusi:** Bagi tugas sesuai kemampuan, adakan sesi belajar bersama, dan gunakan referensi/tutorial.

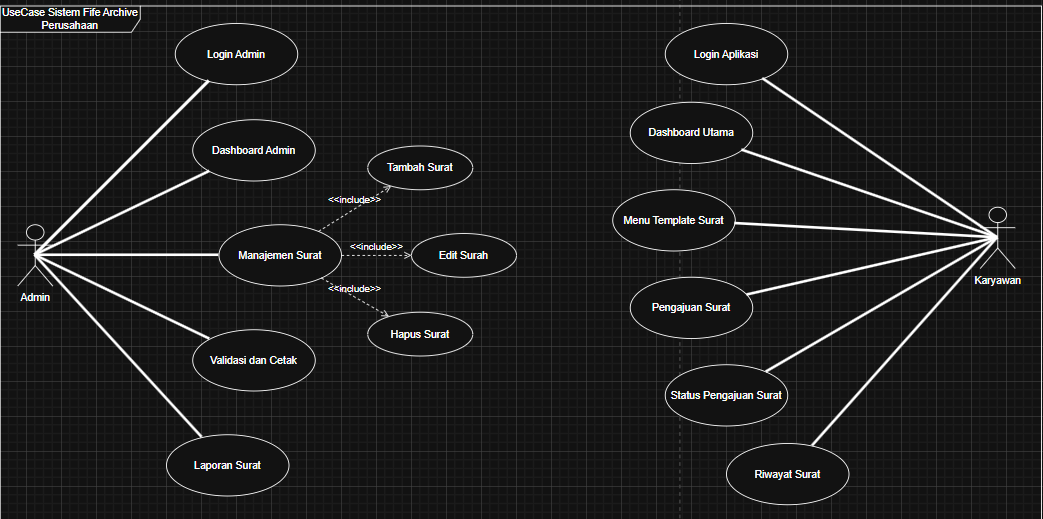
* **Sistem Tidak Sesuai Kebutuhan User**

Sistem bisa meleset dari harapan jika analisis awal tidak akurat.  
**Solusi:** Lakukan wawancara atau observasi kebutuhan user, dan libatkan mereka dalam uji coba awal.

# BAB III ANALISIS DAN PERANCANGAN SISTEM

3.1 Use Case Diagram dan Deskripsinya

* Use Case Diagram Sistem File Archive Perusahaan :



* Deskripsi Login Admin :

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Diagram | : | Login Admin | |
| Triger Event | : | Admin membuka halaman login dan memasukkan kredensial | |
| Aktor | : | Admin | |
| Pre Condition | : | * Admin belum login ke system * Admin berhasil login dan diarahkan ke dashboard admin | |
| Post Condition | : | Data calon peserta didik tersimpan dalam sistem dan siap untuk diproses lebih lanjut. | |
| Normal Course | : | Aktor | Sistem |
|  |  | 1. Memasukkan username dan password | 1. Memverifikasi kredensial |
|  |  | 2. Menekan tombol Login | 2. Mengarahkan ke halaman dashboard jika berhasil |
| Alternative Course | : | Jika login gagal, sistem menampilkan pesan kesalahan | |
| Exception | : | Form kosong atau inputs salah, sistem memberi pesan kesalahan | |

* Deskripsi Dashboard Admin

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Diagram | : | Dashboard Admin | |
| Triger Event | : | Admin berhasil login | |
| Aktor | : | Admin | |
| Pre Condition | : | Admin telah login | |
| Post Condition | : | Admin dapat mengakses fitur-fitur utama | |
| Normal Course | : | Aktor | Sistem |
|  |  | 1. Admin | 2. Menampilkan menu utama dan data dashboard |
|  |  | 2. Menekan tombol Login | 2. Mengarahkan ke halaman dashboard jika berhasil |
| Alternative Course | : | - | |
| Exception | : | - | |

* Deskripsi Manajemen Surat

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Diagram | : | Manajemen Surat | |
| Triger Event | : | Admin memilih menu manajemen surat | |
| Aktor | : | Admin | |
| Pre Condition | : | Admin telah login dan berada di dashboard | |
| Post Condition | : | Data surat berhasil ditambah, diubah, atau dihapus | |
| Normal Course | : | Aktor | Sistem |
|  |  | 1. Memilih menu Manajemen Surat | 1. Menampilkan daftar surat |
|  |  | 2. Memilih Tambah/Edit/Hapus Surat | 2. Menampilkan form |
|  |  | 3. Mengisi form surat dan menyimpan | 3. Menyimpan data surat ke database |
| Alternative Course | : | Admin membatalkan input atau kembali ke daftar surat | |
| Exception | : | Input tidak valid, sistem memberi pesan kesalahan | |

* Deskripsi Validasi dan Cetak

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Diagram | : | Validasi dan Cetak | |
| Triger Event | : | Admin memilih surat untuk divalidasi | |
| Aktor | : | Admin | |
| Pre Condition | : | Surat telah diajukan | |
| Post Condition | : | Surat tervalidasi dan siap cetak | |
| Normal Course | : | Aktor | Sistem |
|  |  | 1. Memilih surat | 1. Menampilkan surat |
|  |  | 2. Memvalidasi dan mencetak | 2. Mengubah status surat |
|  |  |  | 3. Menyediakan file cetakan |
| Alternative Course | : | - | |
| Exception | : | Surat tidak lengkap → sistem menolak validasi | |

* Deskripsi Laporan Surat

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Diagram | : | Laporan surat | |
| Triger Event | : | Admin memilih menu laporan surat | |
| Aktor | : | Admin | |
| Pre Condition | : | Surat tersedia di sistem | |
| Post Condition | : | Laporan surat tampil atau diekspor | |
| Normal Course | : | Aktor | Sistem |
|  |  | 1. Memilih menu laporan | 1. Menampilkan laporan |
|  |  | 2. Menyaring data | 2. Mengekspor ke PDF/Excel |
| Alternative Course | : | - | |
| Exception | : | - | |

* Deskripsi Login Aplikasi (Karyawan)

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Diagram | : | Login aplikasi | |
| Triger Event | : | Karyawan membuka aplikasi | |
| Aktor | : | Karyawan | |
| Pre Condition | : | Belum login | |
| Post Condition | : | Berhasil login dan ke dashboard | |
| Normal Course | : | Aktor | Sistem |
|  |  | 1.Memasukkan data login | 1.Memvertifikasi login |
|  |  |  | 2. Menampilkan dashboard |
| Alternative Course | : | - | |
| Exception | : | Login gagal→ tampilkan pesan error | |

* Deskripsi Dashboard Utama (Karyawan)

|  |  |  |
| --- | --- | --- |
| Use Case Diagram | : | Dashboard utama |
| Triger Event | : | Login berhasil |
| Aktor | : | Karyawan |
| Pre Condition | : | Sudah login |
| Post Condition | : | Dapat mengakses fitur |
| Normal Course | : | Sistem menampilkan menu utama dan info pengguna |
| Alternative Course | : | - |
| Exception | : | - |

* Deskripsi Menu Tempalte Surat

|  |  |  |
| --- | --- | --- |
| Use Case Diagram | : | Menu template surat |
| Triger Event | : | Karyawan memilih template surat |
| Aktor | : | Karyawan |
| Pre Condition | : | Sudah login |
| Post Condition | : | Dapat memilih jenis surat |
| Normal Course | : | Sistem menampilkan daftar template surat |
| Alternative Course | : | - |
| Exception | : | - |

* Deskripsi Pengajuan Surat

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Diagram | : | Pengajuan surat | |
| Triger Event | : | Karyawan memilih template dan mengisi surat | |
| Aktor | : | Karyawan | |
| Pre Condition | : | Sudah login dan memilih template | |
| Post Condition | : | Surat berhasil diajukan | |
| Normal Course | : | Aktor | Sistem |
|  |  | 1.Mengisi form surat | 1.menyimpan data surat ke database |
| Alternative Course | : | - | |
| Exception | : | Input kosong/invalid → tampilkan pesan kesalahan | |

* Deskripsi Status Pengajuan Surat

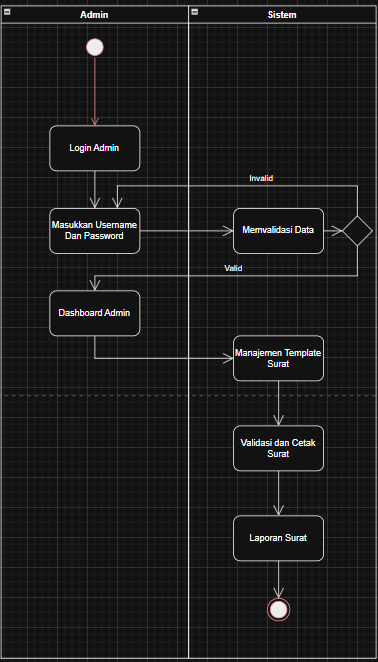
|  |  |  |
| --- | --- | --- |
| Use Case Diagram | : | Status pengajuan surat |
| Triger Event | : | Karyawan membuka menu status |
| Aktor | : | Karyawan |
| Pre Condition | : | Sudah pernah diajukan |
| Post Condition | : | Mengetahui status surat |
| Normal Course | : | Sistem menampilkan status : valid/ditolak/menunggu |
| Alternative Course | : | - |
| Exception | : | - |

* Deskripsi Riwayat Surat

|  |  |  |
| --- | --- | --- |
| Use Case Diagram | : | Riwayat surat |
| Triger Event | : | Karyawan memilih menu riwayat |
| Aktor | : | Karyawan |
| Pre Condition | : | Pernah mengajukan surat |
| Post Condition | : | Dapat melihat surat yang pernah di ajukan |
| Normal Course | : | Sistem menampilkan Riwayat surat berdasarkan user |
| Alternative Course | : | - |
| Exception | : | - |

3.2 Activity Diagram

* Activity Diagram Admin



Penjelasan :

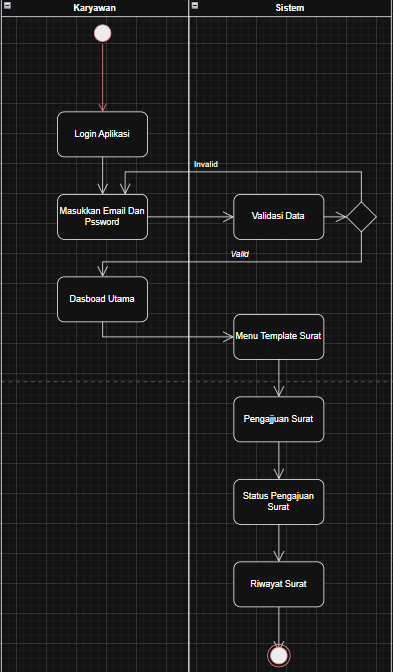
* **Tujuan Activity Diagram**

Diagram ini menunjukkan langkah-langkah aktivitas yang dilakukan oleh Admin dan bagaimana Sistem merespons, dari proses login sampai ke pengelolaan surat.

* **Penjelasan Detail Langkah Aktivitas**

1. **Initial Node**
   * Menandakan awal aktivitas sistem oleh **Admin**.
2. **Login Admin (Admin)**
   * Admin memulai proses dengan memilih menu login.
3. **Masukkan Username dan Password (Admin)**
   * Admin mengisi kredensial untuk mengakses sistem.
4. **Memvalidasi Data (Sistem)**
   * Sistem memproses dan memeriksa apakah username dan password valid.
   * Ada dua kemungkinan:
     + **Valid** → lanjut ke dashboard
     + **Invalid** → kembali ke proses input dengan kemungkinan error message (meskipun tidak digambarkan di diagram, ini umum terjadi).
5. **Dashboard Admin (Admin)**
   * Setelah berhasil login, Admin diarahkan ke halaman dashboard.
   * Di sini Admin dapat memilih berbagai menu yang tersedia.
6. **Manajemen Template Surat (Sistem)**
   * Admin memilih fitur untuk mengelola surat (misalnya tambah/edit/hapus template surat).
   * Sistem memproses dan menampilkan halaman manajemen surat.
7. **Validasi dan Cetak Surat (Sistem)**
   * Setelah surat diajukan oleh karyawan, Admin bisa memvalidasi dan mencetak surat yang disetujui.
8. **Laporan Surat (Sistem)**
   * Admin dapat menghasilkan laporan surat, baik berupa daftar, jumlah, atau rekap surat yang telah dikelola.
9. **Final Node**
   * Menandakan akhir dari aktivitas alur proses admin.

* **Manfaat Activity Diagram Ini**
* **Mempermudah perancangan sistem** sebelum dibangun.
* Memastikan **alur login dan alur utama admin** (manajemen surat, validasi, cetak, laporan) sudah jelas.
* Membantu developer dan tim memahami **urutan proses** dan **respon sistem**.
* Activity Diagram Karyawan



Penjelasan :

* **Tujuan Activity Diagram**

Diagram ini menjelaskan langkah-langkah atau proses bisnis utama yang dilakukan oleh karyawan dalam menggunakan Sistem Arsip Surat Perusahaan. Diagram dibagi menjadi dua bagian (swimlane):

* Karyawan: Menunjukkan aktivitas yang dilakukan pengguna.
* Sistem: Menunjukkan bagaimana sistem merespons aktivitas tersebut.
* **Penjelasan Detail Langkah Aktivitas**

1. Initial Node

* Titik awal aktivitas sistem dimulai saat karyawan membuka aplikasi.

2. Login Aplikasi (Karyawan)

* Karyawan memulai dengan memilih menu login di halaman awal aplikasi.

3. Masukkan Email dan Password (Karyawan)

* Karyawan mengisi form login dengan email dan kata sandi.

4. Validasi Data (Sistem)

* Sistem memproses dan memeriksa apakah data login valid:
  + Jika Valid: lanjut ke halaman dashboard.
  + Jika Tidak Valid: kembali ke proses input dengan kemungkinan pesan error.

5. Dasbor Utama (Karyawan)

* Setelah login berhasil, karyawan diarahkan ke dashboard utama.
* Dari dashboard, karyawan bisa mengakses berbagai fitur aplikasi.

6. Menu Template Surat (Sistem)

* Karyawan memilih jenis template surat yang ingin diajukan dari menu.
* Sistem menampilkan daftar template surat.

7. Pengajuan Surat (Sistem)

* Karyawan mengisi formulir pengajuan berdasarkan template yang dipilih.
* Sistem menyimpan data pengajuan surat.

8. Status Pengajuan Surat (Sistem)

* Karyawan dapat melihat status pengajuan surat:
  + Menunggu verifikasi
  + Disetujui
  + Ditolak

9. Riwayat Surat (Sistem)

* Sistem menyimpan dan menampilkan daftar surat yang pernah diajukan oleh karyawan.

10. Final Node

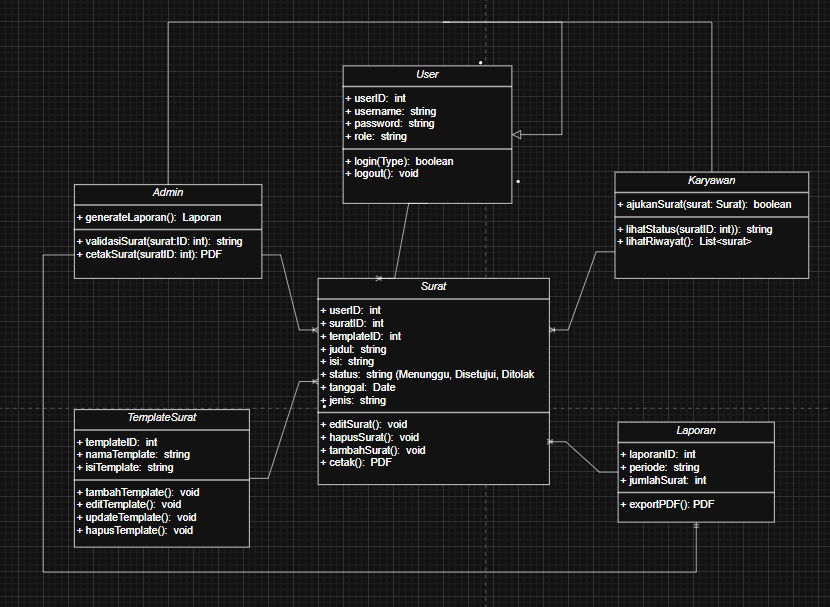
* Titik akhir dari aktivitas setelah karyawan selesai menggunakan fitur-fitur yang tersedia.

Manfaat Diagram Ini

* Memberikan gambaran alur proses karyawan dari login hingga melihat riwayat surat.
* Menunjukkan peran sistem dalam setiap proses (validasi, menampilkan template, menyimpan pengajuan, dll).
* Cocok digunakan untuk dasar pembuatan sistem agar tim pengembang dan stakeholder paham urutan logikanya.

3.3 Class Diagram

* Class Diagram Siatem File Archive Perusahaan



Penjelasan :

**1. User**

* Class dasar untuk semua pengguna.
* **Atribut**:
  + userID: ID unik pengguna.
  + username, password: kredensial login.
  + role: peran pengguna (Admin atau Karyawan).
* **Metode (Fungsi)**:
  + login(Type): proses login ke sistem.
  + logout(): keluar dari sistem.

*Penjelasan*: Class ini bersifat umum, dan akan digunakan oleh class Admin dan Karyawan melalui pewarisan (inheritance).

**2. Admin *(turunan dari User)***

* Digunakan untuk pengguna yang memiliki hak khusus (admin).
* **Metode**:
  + generateLaporan(): membuat laporan surat.
  + validasiSurat(suratID): menyetujui atau menolak surat yang diajukan.
  + cetakSurat(suratID): mencetak surat dalam bentuk PDF.

*Penjelasan*: Admin memiliki akses penuh terhadap data surat dan laporan.

**3. Karyawan *(turunan dari User)***

* Digunakan untuk pegawai yang ingin membuat surat.
* **Metode**:
  + ajukanSurat(surat): mengajukan surat baru.
  + lihatStatus(suratID): melihat status dari surat yang diajukan.
  + lihatRiwayat(): melihat daftar surat yang pernah diajukan.

*Penjelasan*: Karyawan hanya memiliki akses terbatas pada surat miliknya sendiri.

**4. Surat**

* Mewakili dokumen surat yang dikelola oleh sistem.
* **Atribut**:
  + userID: pemilik surat.
  + suratID: ID unik surat.
  + templateID: jenis template surat yang digunakan.
  + judul, isi, jenis: isi konten surat.
  + status: status surat (Menunggu, Disetujui, Ditolak).
  + tanggal: tanggal surat dibuat.
* **Metode**:
  + editSurat(), hapusSurat(), tambahSurat(): aksi pada surat.
  + cetak(): mencetak surat ke format PDF.

*Penjelasan*: Surat berelasi dengan User (pembuat) dan TemplateSurat (jenis surat).

**5. TemplateSurat**

* Template atau bentuk dasar surat yang bisa digunakan.
* **Atribut**:
  + templateID, namaTemplate, isiTemplate
* **Metode**:
  + tambahTemplate(), updateTemplate(), editTemplate(), hapusTemplate()

*Penjelasan*: Admin bisa membuat template ini agar karyawan bisa memilih saat mengajukan surat.

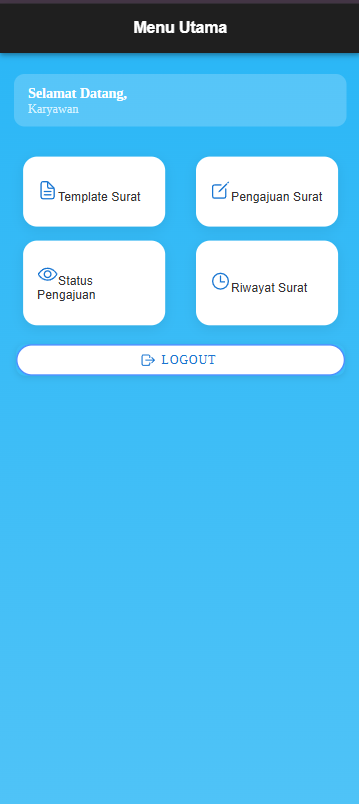
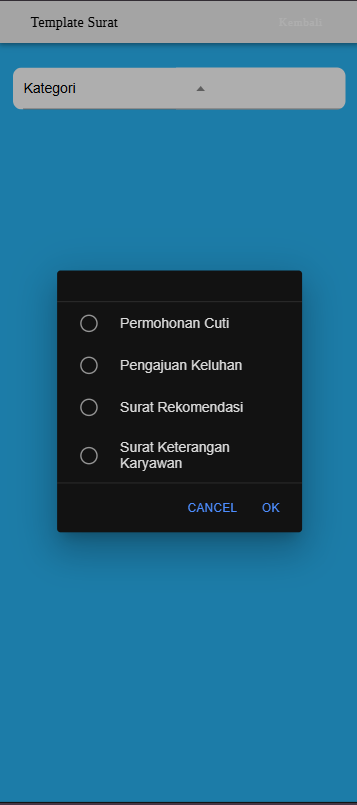
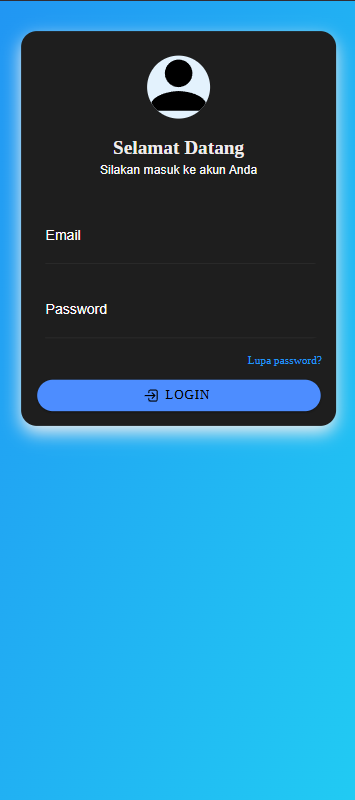
**6. Laporan**

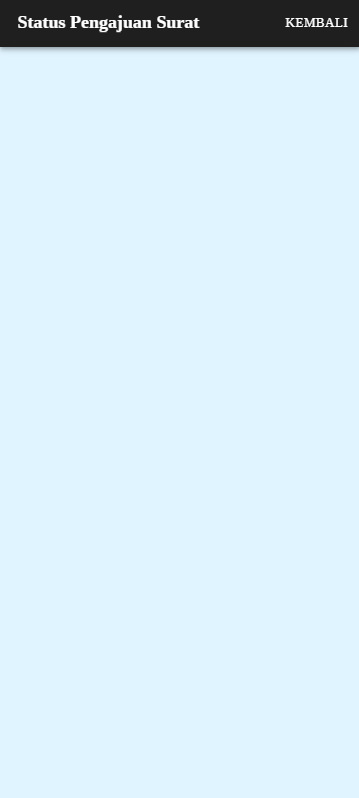
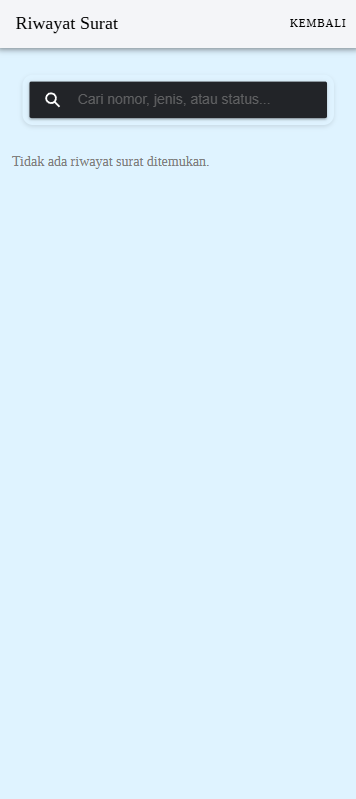
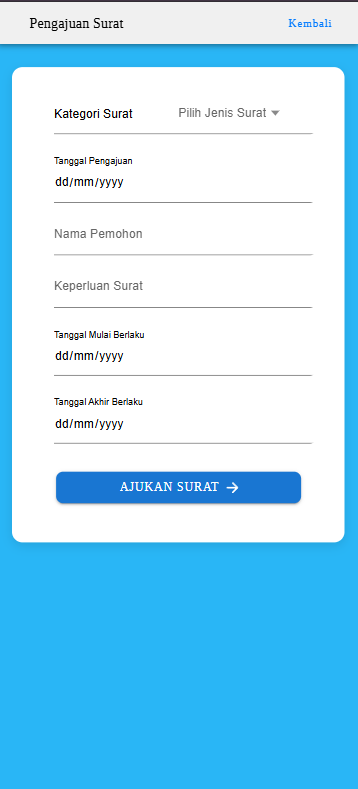
* Berisi rekap surat dalam suatu periode.
* **Atribut**:
  + laporanID, periode, jumlahSurat
* **Metode**:
  + exportPDF(): mengekspor laporan ke PDF.

*Penjelasan*: Digunakan untuk keperluan pelaporan yang dilakukan oleh Admin.

3.4 Mockup UI/UX

3.4.1 Mockup UI/UX Versi Mobile

****

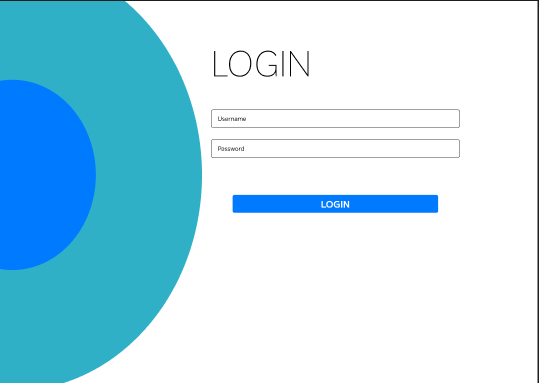
****

* **Penjelasan UI Mockup Aplikasi Arsip Surat (Versi Mobile)**

1. **Halaman Login**
   * Menyediakan kolom input untuk **Email** dan **Password**.
   * Tersedia tombol **Login** dan link **Lupa password?** untuk mengatur ulang kata sandi.
   * Tampilan bersih dengan latar gradasi biru, serta ikon pengguna di bagian atas.
2. **Menu Utama**
   * Menampilkan **ucapan selamat datang** beserta peran pengguna (misalnya "Karyawan").
   * Terdapat empat menu utama: Template Surat, Pengajuan Surat, Status Pengajuan, dan Riwayat Surat.
   * Tersedia tombol **Logout** di bagian bawah.
   * Desain minimalis dan intuitif dengan ikon untuk tiap menu.
3. **Template Surat**
   * Menampilkan dropdown kategori surat.
   * Saat diklik, muncul pilihan jenis surat seperti: Permohonan Cuti, Pengajuan Kegiatan, Surat Rekomendasi, dan Surat Keterangan Karyawan.
   * User memilih template sebelum membuat pengajuan.
4. **Pengajuan Surat**
   * Formulir pengisian surat berdasarkan template yang dipilih.
   * Input meliputi: Tanggal pengajuan, Nama, Keperluan, Tanggal mulai dan akhir.
   * Dilengkapi tombol **Ajukan Surat** di bagian bawah.
5. **Status Pengajuan Surat**
   * Menampilkan daftar status surat yang telah diajukan.
   * Setiap surat akan ditampilkan dengan status seperti “Diproses”, “Disetujui”, atau “Ditolak”.
   * Terdapat tombol **Kembali** di header.
6. **Riwayat Surat**
   * Menampilkan seluruh surat yang pernah diajukan oleh pengguna.
   * Terdapat fitur **pencarian** untuk memfilter berdasarkan nama, jenis, atau tanggal.
   * Jika belum ada data, ditampilkan pesan **"Tidak ada riwayat surat ditemukan"**.
   * Header juga dilengkapi tombol **Kembali**.

### 3.4.2 Mockup UI/UX Tampilan Website

* Tampilan Login



Halaman *Login* pada tampilan website dirancang dengan pendekatan minimalis dan modern untuk memberikan kesan profesional sejak awal pengguna mengakses sistem. Antarmuka terdiri atas dua bidang isian utama, yaitu **username** dan **password**, yang disusun secara vertikal dan sejajar di bagian tengah halaman.

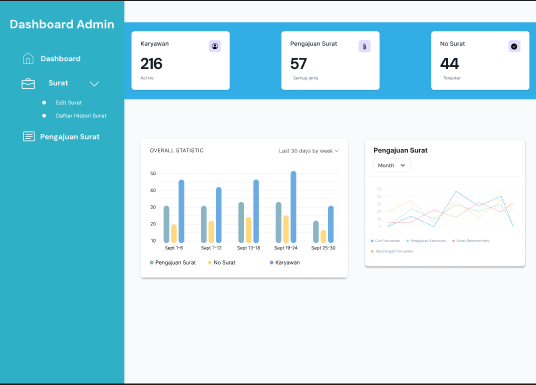
Tombol **LOGIN** ditampilkan dengan warna biru solid, ditempatkan tepat di bawah kedua kolom input sebagai aksi utama. Penggunaan warna biru bertujuan untuk menarik perhatian pengguna terhadap tindakan yang harus dilakukan.

Sebagai elemen visual tambahan, terdapat dekorasi grafis berupa **lingkaran konsentris biru dan toska** pada sisi kiri layar. Elemen ini tidak hanya memperkuat estetika halaman, tetapi juga memberikan identitas visual yang khas terhadap sistem.

Secara keseluruhan, halaman ini dirancang untuk:

1. Memastikan alur login yang sederhana dan intuitif
2. Menjaga keseimbangan estetika dan fungsionalitas
3. Meningkatkan pengalaman pengguna dengan desain yang bersih dan responsif

* Tampilan Dashboard Admin



Halaman *Dashboard Admin* berfungsi sebagai pusat kendali utama bagi administrator dalam memantau aktivitas dan data penting dalam sistem pengelolaan arsip surat. Antarmuka halaman ini disusun secara informatif dan responsif, dengan memprioritaskan visualisasi data statistik serta navigasi menu yang terstruktur.

Pada bagian kiri layar, terdapat **panel navigasi vertikal** dengan latar belakang biru muda yang memuat menu utama, seperti:

* Dashboard
* Surat (dengan submenu: Edit Surat dan Daftar Histori Surat)
* Pengajuan Surat

Sementara itu, bagian utama dashboard menampilkan tiga **kartu informasi ringkas (summary box)** yang menunjukkan data sebagai berikut:

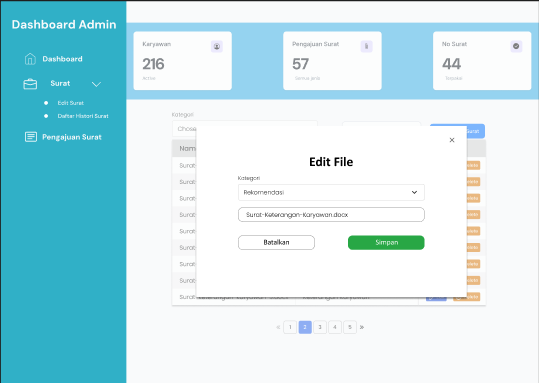
* Jumlah Karyawan: 216
* Jumlah Pengajuan Surat: 57
* Jumlah Nomor Surat Terdaftar: 44

Di bawahnya, terdapat dua grafik interaktif:

* **Grafik batang (bar chart)** yang menunjukkan statistik pengajuan surat, jumlah nomor surat, dan data karyawan dalam kurun waktu mingguan.
* **Grafik garis (line chart)** yang menampilkan tren pengajuan surat berdasarkan bulan.

Desain halaman ini mengutamakan:

* Keterbacaan data dengan kontras warna biru dan putih
* Penyajian informasi real-time dan terpusat
* Kemudahan akses menu melalui struktur yang jelas
* Tampilan Edit Surat



Halaman *Edit Surat* merupakan bagian dari panel administrasi yang memberikan akses kepada admin untuk mengelola data template surat yang tersedia dalam sistem. Antarmuka halaman ini menggabungkan fungsi tampilan daftar dan formulir interaktif dalam satu tampilan yang ringkas dan efisien.

Di latar belakang, ditampilkan **tabel daftar surat** yang berisi informasi-informasi penting seperti nama file, kategori surat, serta aksi yang dapat dilakukan. Setiap baris dilengkapi dengan tombol aksi seperti edit, hapus, dan pratinjau (*preview*), untuk mempermudah pengelolaan file.

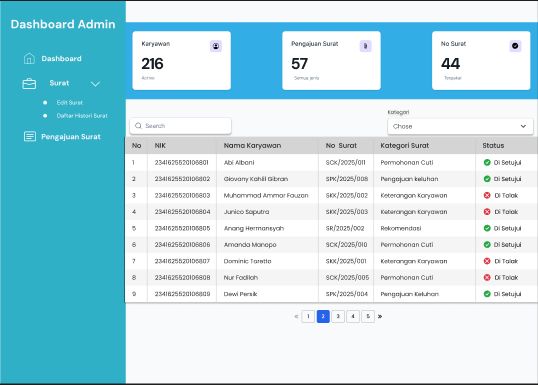
Ketika tombol **Edit** ditekan, akan muncul **modal popup** di bagian tengah layar dengan judul “Edit File”. Pada jendela ini, admin dapat:

* Memilih kategori surat melalui menu dropdown
* Mengubah nama file surat atau judul deskriptif surat
* Menyimpan perubahan dengan menekan tombol **Simpan**
* Membatalkan proses melalui tombol **Batal**

Tampilan popup ini bersifat intuitif dan fokus, menghindari gangguan dari elemen lain di halaman. Elemen-elemen visual seperti dropdown, tombol aksi berwarna hijau, serta layout form dua kolom dirancang untuk mempercepat interaksi dan meminimalisasi kesalahan input.

Desain halaman ini secara keseluruhan mendukung:

* Pengelolaan template surat secara cepat dan akurat
* Navigasi yang terstruktur dalam manajemen data
* Penggunaan komponen visual yang konsisten dengan antarmuka lainnya
* Tampilan History Surat



Halaman Histori Surat merupakan tampilan yang digunakan oleh admin untuk memantau seluruh data pengajuan surat yang telah masuk ke dalam sistem. Tampilan ini disusun dalam bentuk tabel yang rapi dan responsif, memuat informasi penting seperti:

* Nomor Induk Karyawan (NIK)
* Nama Karyawan
* Nomor Surat
* Kategori Surat
* Status Surat (disetujui, ditolak, atau menunggu verifikasi)

Di bagian atas halaman, ditampilkan tiga kotak ringkasan (summary box) yang menunjukkan total jumlah karyawan, total pengajuan surat, dan total nomor surat yang terdaftar. Informasi ini memberikan gambaran cepat terhadap aktivitas surat-menyurat di perusahaan.

Fitur pencarian disediakan dalam bentuk kolom teks dan dropdown kategori surat untuk memudahkan admin dalam menyaring data berdasarkan jenis surat yang ditampilkan. Hal ini meningkatkan efisiensi pencarian surat dalam jumlah data yang besar.

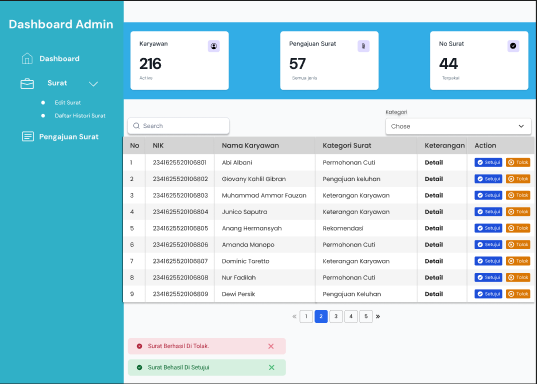
Sementara itu, indikator status surat ditampilkan dengan ikon berwarna:

* Hijau: Surat disetujui
* Merah: Surat ditolak
* Kuning: Surat sedang diproses atau menunggu verifikasi

Tabel histori juga dilengkapi dengan navigasi halaman (pagination) di bagian bawah agar admin dapat menelusuri seluruh data surat secara bertahap.

Dengan demikian, halaman ini dirancang untuk mempermudah pengelolaan dokumen administrasi secara cepat, informatif, dan terstruktur.

* Tampilan Pengajuan Surat



Halaman Pengajuan Surat merupakan bagian penting dari sistem yang berfungsi sebagai pusat validasi seluruh permintaan surat yang diajukan oleh pengguna. Tampilan ini dirancang untuk admin agar dapat melihat daftar pengajuan secara menyeluruh, serta memproses setiap permohonan berdasarkan status dan kelengkapannya.

Antarmuka halaman terdiri atas tabel daftar pengajuan yang menampilkan kolom-kolom berikut:

* NIK (Nomor Induk Karyawan)
* Nama Karyawan
* Kategori Surat
* Keterangan
* Aksi

Setiap baris dilengkapi dengan tombol aksi berupa:

* Detail (warna biru): Menampilkan informasi lengkap dari pengajuan surat.
* Tolak (warna merah): Menolak surat yang tidak sesuai atau tidak lengkap.
* Setujui (warna hijau): Menyetujui surat yang valid dan sesuai persyaratan.

Di atas tabel, terdapat fitur pencarian teks dan filter dropdown berdasarkan kategori surat, yang mempermudah admin dalam menemukan pengajuan tertentu dengan cepat dan efisien.

Pada bagian bawah halaman, terdapat kotak informasi notifikasi yang memberikan umpan balik kepada admin mengenai status keputusan yang telah diambil, seperti:

* Surat berhasil ditolak (latar merah muda dengan ikon silang)
* Surat berhasil disetujui (latar hijau muda dengan ikon centang)

Elemen visual ini membantu admin mengetahui hasil aksi mereka secara langsung tanpa perlu berpindah halaman.

Desain halaman secara keseluruhan mengedepankan kemudahan penggunaan, kejelasan data, serta efisiensi proses verifikasi pengajuan surat di lingkungan perusahaan.

3.5 Rancangan Basis Data

### 3.5.1 Penjelasan Struktur Database

Struktur database dalam sistem arsip surat ini dirancang untuk mendukung pengelolaan surat secara digital, mulai dari pembuatan template, pengajuan surat oleh pengguna (karyawan), hingga proses persetujuan dan penyimpanan historis surat yang telah diterbitkan.

Database menggunakan pendekatan relasional, dengan **entitas utama** meliputi:

* **users** (pengguna sistem: admin/karyawan)
* **karyawan** (data pribadi pegawai)
* **template\_surat** (format dasar surat)
* **pengajuan\_surat** (permintaan surat)
* **histori\_surat** (arsip surat yang telah diterbitkan)
* **kategori\_template** (pengelompokan template)
* **log\_nomor\_surat** (penomoran otomatis surat)
* **password\_reset\_tokens** (reset password)

Setiap entitas saling berhubungan dengan **relasi satu-ke-banyak** maupun **satu-ke-satu**, dan didukung oleh foreign key untuk menjaga integritas data.

### 3.5.2 Tabel-Tabel Utama dan Atributnya

Berikut adalah ringkasan dari tabel-tabel utama yang digunakan dalam sistem:

* **users**

Menyimpan data akun pengguna sistem (admin atau karyawan).

| **Kolom** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | bigint unsigned | Primary Key |
| nama | varchar(255) | Nama pengguna |
| email | varchar(255) | Email login |
| password | varchar(255) | Password terenkripsi |
| role | enum('karyawan','admin') | Peran pengguna |
| created\_at | timestamp | Tanggal dibuat |
| updated\_at | timestamp | Tanggal diperbarui |

* **karyawan**

Berisi data pribadi karyawan yang terkait dengan akun di tabel users.

| **Kolom** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | bigint unsigned | Primary Key |
| user\_id | bigint unsigned | FK ke users.id |
| nik | varchar(255) | Nomor Induk Karyawan |
| nama\_lengkap | varchar(255) | Nama lengkap |
| email | varchar(255) | Email karyawan |
| created\_at | timestamp |  |
| updated\_at | timestamp |  |

* **template\_surat**

Berisi template atau format surat yang bisa digunakan untuk pengajuan.

| **Kolom** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | bigint unsigned | Primary Key |
| judul\_template | varchar(255) | Judul atau nama template |
| file\_path | varchar(255) | Path file template (.docx/.pdf) |
| format\_file | enum('pdf','docx') | Format file template |
| kategori\_id | bigint unsigned | FK ke kategori\_template.id |
| created\_at | timestamp |  |
| updated\_at | timestamp |  |

* **kategori\_template**

Berisi kategori surat, seperti “Surat Keterangan”, “Surat Cuti”, dll.

| **Kolom** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | bigint unsigned | Primary Key |
| nama\_kategori | varchar(255) | Nama kategori template |

* **pengajuan\_surat**

Mewakili pengajuan surat oleh karyawan.

| **Kolom** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | bigint unsigned | Primary Key |
| user\_id | bigint unsigned | FK ke users.id |
| template\_id | bigint unsigned | FK ke template\_surat.id |
| tanggal\_pengajuan | date | Tanggal surat diajukan |
| keterangan | text | Catatan atau alasan pengajuan |
| status | enum('menunggu') | Status awal pengajuan |
| tanggal\_berlaku | date | Tanggal mulai berlaku (opsional) |
| tanggal\_berakhir | date | Tanggal selesai berlaku (opsional) |
| created\_at | timestamp |  |
| updated\_at | timestamp |  |

* **histori\_surat**

Berisi arsip surat yang telah diterbitkan dan disetujui.

| **Kolom** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | bigint unsigned | Primary Key |
| user\_id | bigint unsigned | FK ke users.id |
| template\_id | bigint unsigned | FK ke template\_surat.id |
| nomor\_surat | varchar(255) | Nomor resmi surat yang diterbitkan |
| kategori\_id | bigint unsigned | FK ke kategori\_template.id |
| keterangan | text | Catatan admin/pemberi surat |
| file\_hasil\_path | varchar(255) | Lokasi file final surat (PDF) |
| tanggal\_diterbitkan | date | Tanggal surat diterbitkan |
| status | enum('selesai','ditolak','dibatalkan') | Status surat |
| created\_at | timestamp |  |
| updated\_at | timestamp |  |

* **log\_nomor\_surat**

Digunakan untuk pencatatan penomoran surat otomatis berdasarkan tahun dan kategori.

| **Kolom** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | bigint unsigned | Primary Key |
| kategori\_id | bigint unsigned | FK ke kategori\_template.id |
| tahun | int | Tahun surat diterbitkan |
| last\_number | int | Nomor terakhir yang digunakan |
| created\_at | timestamp |  |
| updated\_at | timestamp |  |

* **password\_reset\_tokens**

Menyimpan token reset password untuk sistem autentikasi.

| **Kolom** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| email | varchar(255) | Email pengguna |
| token | varchar(255) | Token reset password |
| created\_at | timestamp |  |

### 3.5.3 ERD (Entity Relationship Diagram)

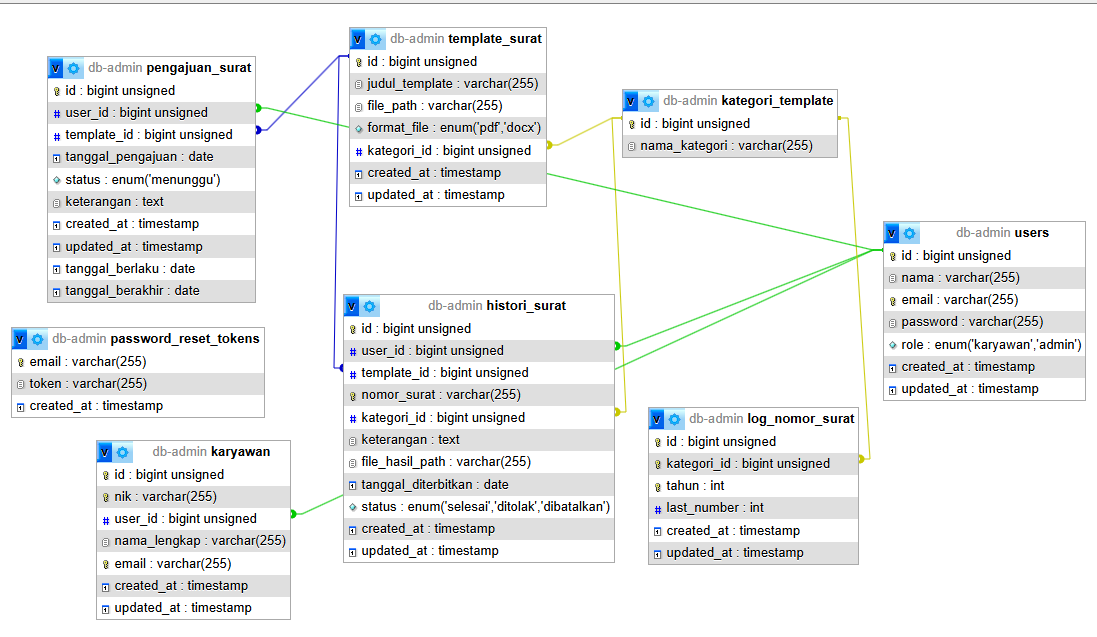


Diagram yang kamu lampirkan menunjukkan hubungan antara tabel-tabel dengan sangat jelas, termasuk foreign key dan jenis relasinya. Berikut ini rangkuman relasi:

* users → karyawan (1-to-1)
* users → pengajuan\_surat, histori\_surat (1-to-many)
* template\_surat → pengajuan\_surat, histori\_surat (1-to-many)
* kategori\_template → template\_surat, log\_nomor\_surat, histori\_surat (1-to-many)

ERD ini menjadi representasi visual dari desain database relasional, dan penting sebagai dokumentasi pengembangan sistem.

### 3.5.4 Skrip Sql

CREATE TABLE `users` (

  `id` bigint UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY,

  `nama` varchar(255) NOT NULL,

  `email` varchar(255) NOT NULL UNIQUE,

  `password` varchar(255) NOT NULL,

  `role` enum('karyawan','admin') NOT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

);

CREATE TABLE `karyawan` (

  `id` bigint UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY,

  `nik` varchar(255) NOT NULL UNIQUE,

  `user\_id` bigint UNSIGNED NOT NULL,

  `nama\_lengkap` varchar(255) NOT NULL,

  `email` varchar(255) NOT NULL UNIQUE,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL,

  FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ON DELETE CASCADE

);

CREATE TABLE `kategori\_template` (

  `id` bigint UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY,

  `nama\_kategori` varchar(255) NOT NULL

);

CREATE TABLE `template\_surat` (

  `id` bigint UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY,

  `judul\_template` varchar(255) NOT NULL,

  `file\_path` varchar(255) NOT NULL,

  `format\_file` enum('pdf','docx') NOT NULL,

  `kategori\_id` bigint UNSIGNED NOT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL,

  FOREIGN KEY (`kategori\_id`) REFERENCES `kategori\_template` (`id`) ON DELETE CASCADE

);

CREATE TABLE `pengajuan\_surat` (

  `id` bigint UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY,

  `user\_id` bigint UNSIGNED NOT NULL,

  `template\_id` bigint UNSIGNED NOT NULL,

  `tanggal\_pengajuan` date NOT NULL,

  `status` enum('menunggu') NOT NULL DEFAULT 'menunggu',

  `keterangan` text,

  `tanggal\_berlaku` date DEFAULT NULL,

  `tanggal\_berakhir` date DEFAULT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL,

  FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ON DELETE CASCADE,

  FOREIGN KEY (`template\_id`) REFERENCES `template\_surat` (`id`) ON DELETE CASCADE

);

CREATE TABLE `histori\_surat` (

  `id` bigint UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY,

  `user\_id` bigint UNSIGNED NOT NULL,

  `template\_id` bigint UNSIGNED NOT NULL,

  `nomor\_surat` varchar(255) NOT NULL UNIQUE,

  `kategori\_id` bigint UNSIGNED NOT NULL,

  `keterangan` text,

  `file\_hasil\_path` varchar(255) DEFAULT NULL,

  `tanggal\_diterbitkan` date DEFAULT NULL,

  `status` enum('selesai','ditolak','dibatalkan') NOT NULL DEFAULT 'selesai',

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL,

  FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ON DELETE CASCADE,

  FOREIGN KEY (`template\_id`) REFERENCES `template\_surat` (`id`) ON DELETE CASCADE,

  FOREIGN KEY (`kategori\_id`) REFERENCES `kategori\_template` (`id`) ON DELETE CASCADE

);

CREATE TABLE `log\_nomor\_surat` (

  `id` bigint UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY,

  `kategori\_id` bigint UNSIGNED NOT NULL,

  `tahun` int NOT NULL,

  `last\_number` int NOT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL,

  UNIQUE (`kategori\_id`, `tahun`),

  FOREIGN KEY (`kategori\_id`) REFERENCES `kategori\_template` (`id`) ON DELETE CASCADE

);

CREATE TABLE `password\_reset\_tokens` (

  `email` varchar(255) NOT NULL PRIMARY KEY,

  `token` varchar(255) NOT NULL,

  `created\_at` timestamp NULL DEFAULT NULL

);

3.6 Perancangan API

Berisi rancangan antarmuka backend (API) yang akan digunakan, seperti endpoint, method, parameter, dan response. Jika menggunakan framework tertentu (seperti Laravel, Express, dsb.), cantumkan struktur route-nya.

### 3.6.1 Struktur Route di Laravel (routes/api.php)

<?php

use App\Http\Controllers\AuthController;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Route;

use App\Http\Controllers\Api\TemplateSuratApiController;

use App\Http\Controllers\Api\PengajuanSuratApiController;

use App\Http\Controllers\Api\RiwayatSuratApiController;

/\*

|--------------------------------------------------------------------------

| API Routes

|--------------------------------------------------------------------------

|

| Here is where you can register API routes for your application. These

| routes are loaded by the RouteServiceProvider and all of them will

| be assigned to the "api" middleware group. Make something great!

|

\*/

Route::middleware('auth:api')->get('/pengajuan/status', [PengajuanSuratApiController::class, 'getStatusPengajuan']);

Route::middleware('auth:api')->get('/pengajuan/riwayat', [\App\Http\Controllers\Api\RiwayatSuratApiController::class, 'getRiwayatUser']);

Route::middleware('auth:api')->get('/me', function (Request $request) {

    return $request->user();

});

Route::get('/template-surat', [TemplateSuratApiController::class, 'index']);

Route::post('/pengajuan-surat', [PengajuanSuratApiController::class, 'store']);

Route::post('/login', [AuthController::class, 'apiLogin']);

Route::middleware('auth:api')->post('/logout', function () {

    auth()->logout();

    return response()->json(['message' => 'Successfully logged out']);

});

### 3.6.2 Dokumentasi API Laravel (Endpoint, Method, Parameter, Response)

| **Endpoint** | **Method** | **Deskripsi** | **Parameter Body / Query** | **Auth?** |
| --- | --- | --- | --- | --- |
| /login | POST | Login dan dapatkan token | email, password | ✅ |
| /user | GET | Mendapatkan data user login | - | ✅ |
| /logout | POST | Logout dan hapus token | - | ✅ |
| /template-surat | GET | List semua template surat | - | ✅ |
| /template-surat | POST | Tambah template surat baru | judul\_template, kategori\_id, file\_path, format\_file | ✅ |
| /pengajuan-surat | GET | List pengajuan milik user | - | ✅ |
| /pengajuan-surat | POST | Ajukan surat baru | template\_id, tanggal\_pengajuan, keterangan, tanggal\_berlaku, tanggal\_berakhir | ✅ |
| /status-pengajuan | GET | Lihat status pengajuan terakhir | - | ✅ |
| /riwayat-surat | GET | Lihat histori surat user | - | ✅ |

# BAB IV IMPLEMENTASI SISTEM

4.1 Teknologi yang Digunakan

Pengembangan sistem aplikasi arsip surat berbasis web dan mobile ini menggunakan berbagai teknologi modern yang dipilih berdasarkan kestabilan, kemudahan pengembangan, serta dukungan komunitas yang luas. Adapun daftar teknologi yang digunakan adalah sebagai berikut:

* Bahasa Pemrograman
* **PHP**Digunakan untuk membangun backend karena memiliki dokumentasi lengkap dan terintegrasi baik dengan framework Laravel.
* **TypeScript & JavaScript**Digunakan untuk frontend mobile melalui framework Angular, serta mendukung pengembangan antarmuka yang interaktif dan dinamis.
* Framework
* **Laravel**Laravel dipilih sebagai kerangka kerja backend karena menyediakan struktur pengembangan yang rapi, fitur autentikasi bawaan, serta efisiensi dalam membangun RESTful API.
* **Ionic + Angular**Digunakan untuk membangun aplikasi mobile hybrid yang kompatibel dengan berbagai platform. Ionic memungkinkan pengembangan antarmuka berbasis HTML/CSS/JS dengan performa mendekati aplikasi native.
* Database
* **MySQL**

Digunakan sebagai sistem manajemen basis data karena memiliki kinerja tinggi, bersifat open-source, dan mudah terintegrasi dengan Laravel. Struktur relasionalnya sesuai untuk kebutuhan sistem pengarsipan yang kompleks.

* Tools dan Pendukung
* **Visual Studio Code**Digunakan sebagai *code editor* utama karena ringan, fleksibel, dan mendukung berbagai ekstensi pengembangan Laravel maupun Ionic.
* **XAMPP / Laragon**Digunakan untuk menjalankan server lokal selama proses pengembangan backend.
* **Postman**Digunakan untuk menguji endpoint API yang dibangun, serta memverifikasi format request dan response.
* **Git & GitHub**Digunakan sebagai sistem kontrol versi untuk kolaborasi tim dan manajemen versi proyek secara terpusat.
* **Figma**Digunakan untuk merancang antarmuka pengguna (*user interface design*) sebelum proses implementasi dilakukan.
* Alasan Pemilihan

Pemilihan teknologi di atas didasarkan pada pertimbangan kemudahan integrasi, dokumentasi yang lengkap, serta kecocokan dengan kebutuhan akademik dan industri. Kombinasi Laravel dan Ionic memberikan fleksibilitas dalam membangun aplikasi lintas platform dengan waktu pengembangan yang efisien tanpa mengorbankan performa dan skalabilitas sistem.

4.2 Implementasi Fitur-Fitur Utama

Implementasi fitur pada sistem arsip surat digital dilakukan secara modular, di mana setiap fungsi utama dikembangkan sesuai dengan kebutuhan pengguna dan tujuan sistem. Beberapa fitur inti yang diimplementasikan meliputi autentikasi pengguna, pengelolaan template surat, pengajuan surat, validasi surat oleh admin, serta pelacakan history pengajuan. Seluruh fitur dirancang untuk dapat diakses baik melalui aplikasi mobile maupun panel web admin secara responsif dan efisien.

### Fitur Login

Fitur login diimplementasikan menggunakan autentikasi berbasis token (*Laravel Sanctum*). Pengguna (admin maupun karyawan) melakukan login dengan memasukkan email dan kata sandi. Jika berhasil, sistem akan memberikan token akses yang digunakan untuk otorisasi pada permintaan API selanjutnya. Validasi input dilakukan pada sisi frontend dan backend untuk menjamin keamanan data.

### Fitur CRUD Template Surat

Admin dapat menambahkan, mengedit, dan menghapus template surat yang tersedia dalam sistem. Fungsi ini dihubungkan langsung dengan database dan mendukung pengunggahan file dengan format .docx. File yang diunggah disimpan di direktori server, sedangkan metadata-nya disimpan dalam tabel surat. Aksi edit dan hapus dilengkapi dengan konfirmasi dialog untuk menghindari kesalahan manipulasi data.

### Fitur Pengajuan Surat

Karyawan dapat memilih jenis surat berdasarkan kategori, mengisi data keperluan, dan mengajukan permohonan melalui aplikasi. Data yang dikirim akan disimpan dalam tabel pengajuan surat dengan status awal “Menunggu”. Proses ini dilakukan melalui API endpoint /api/pengajuan yang memverifikasi validitas data sebelum menyimpannya ke basis data.

### Fitur Validasi Surat oleh Admin

Admin memiliki akses ke halaman pengajuan surat untuk memverifikasi permintaan yang masuk. Admin dapat meninjau detail surat, kemudian memilih untuk menyetujui atau menolak pengajuan dengan memberikan catatan atau alasan. Keputusan akan tercatat dalam sistem dan status pengajuan diperbarui secara otomatis di dashboard karyawan.

### Fitur Riwayat dan Status Surat

Setiap pengguna dapat melihat riwayat surat yang telah diajukan beserta status terkininya. Fitur ini ditampilkan dalam bentuk daftar atau tabel dengan filter pencarian berdasarkan kategori dan tanggal. Informasi yang ditampilkan meliputi nomor surat, jenis surat, tanggal pengajuan, serta status (Disetujui, Ditolak, atau Menunggu).

4.3 Implementasi Backend dan API

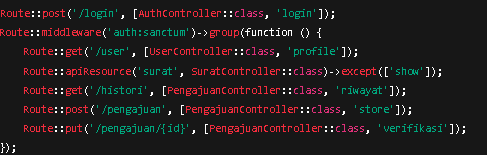
Implementasi backend pada sistem ini memanfaatkan **Laravel** sebagai kerangka kerja utama untuk membangun RESTful API, manajemen autentikasi, dan orkestrasi logika bisnis. Berikut uraian komponen dan alur kerjanya:

* Struktur Proyek dan Pengelompokan Controller

Direktori **app/Http/Controllers/** menyimpan kontroler yang mengolah setiap permintaan (request) dan menyiapkan balasan (response):

* **AuthController**Bertanggung jawab atas autentikasi, termasuk verifikasi kredensial, pembuatan token akses melalui *Laravel Sanctum*, serta proses *logout*.
* **UserController**Menyediakan endpoint untuk memperoleh data profil pengguna yang sedang aktif.
* **SuratController**Menangani operasi CRUD template surat— index(), store(), update(), dan destroy()—serta validasi dan penyimpanan berkas .docx dengan penamaan unik.
* **PengajuanController**Mengelola alur pengajuan surat, meliputi store() untuk menyimpan permohonan baru (status “Menunggu”), index() untuk daftar pengajuan admin, riwayat() untuk histori karyawan, dan verifikasi() untuk memperbarui status pengajuan (Disetujui/Ditolak) beserta catatan verifikasi.
* Definisi Rute dan Perlindungan Middleware

Semua endpoint API didefinisikan di **routes/api.php** dengan konvensi RESTful. Grup rute yang memerlukan autentikasi dijaga oleh middleware auth:sanctum, sehingga hanya pengguna terverifikasi yang dapat mengakses sumber daya selain /login. Contoh deklarasi rute inti:



* Pengelolaan Data dan Logika Bisnis
* **Model Eloquent dan Relasi**Setiap kontroler berinteraksi dengan model Eloquent (User, Surat, Pengajuan Surat, Status Pengajuan, Kategori Surat) untuk menjalankan query sesuai relasi one‑to‑many dan many‑to‑one.
* **Validasi Permintaan**Permintaan POST dan PUT divalidasi menggunakan FormRequest atau metode validate(). Kegagalan validasi menghasilkan respons kode HTTP 422 beserta detail kesalahan dalam format JSON.
* **Penyimpanan Berkas**File .docx disimpan pada storage/app/public/surat dengan penamaan hash untuk mencegah konflik. Jalur berkas disimpan sebagai atribut pada entitas Surat.
* **Logging dan Notifikasi**Setiap perubahan status pengajuan dicatat dalam log aplikasi. Ke depannya, mekanisme ini dapat diperluas untuk mengirim notifikasi email atau *push notification* pada aplikasi mobile.
* Aspek Keamanan dan Skalabilitas
* **Autentikasi dan Otorisasi**Laravel Sanctum menyediakan token berbasis *cookie* atau *bearer token*, memisahkan sesi web dan API.
* **Pembatasan Permintaan (Rate Limiting)**API dilindungi oleh *rate limiter* bawaan Laravel untuk mencegah penyalahgunaan, khususnya pada endpoint sensitif seperti /login.
* **Pemrosesan Asinkron**Proses berat (misalnya, generasi laporan histori panjang) dapat di-*offload* ke antrian (*queues*) untuk menjaga responsivitas API.

Pelaksanaan arsitektur ini memastikan aliran data antar lapisan sistem berjalan konsisten, aman, dan mudah dipelihara, sehingga siap mendukung pengembangan fitur lanjutan di masa depan.

4.4 Implementasi Aplikasi Mobile

Aplikasi mobile dikembangkan menggunakan **Ionic** dengan **Angular** sebagai kerangka kerja utama, sehingga menghasilkan aplikasi hybrid yang dapat berjalan pada platform **Android** dan **iOS**. Pemilihan Ionic didasarkan pada kemudahan integrasi dengan API Laravel dan kemampuan mengemas antarmuka berbasis web menjadi aplikasi seluler yang responsif.

* Arsitektur dan Struktur Proyek
* Direktori **src/app/** berisi komponen, layanan (*services*), dan modul yang dipisahkan sesuai fitur.
* Setiap halaman (*page*) menggunakan *lazy loading* untuk mempercepat waktu muat awal aplikasi.
* Autentikasi dan Manajemen Sesi
* Fitur Login memanfaatkan endpoint /api/login pada backend. Setelah berhasil, token JWT disimpan di Storage lokal (Ionic Storage) untuk otorisasi setiap permintaan API selanjutnya.
* Mekanisme Guard pada route Angular memastikan hanya pengguna terverifikasi yang dapat mengakses halaman internal.
* Fitur Utama
* **Daftar Template Surat**Menampilkan daftar template surat yang diambil dari endpoint /api/surat. Pengguna dapat mengunduh file .docx melalui tautan yang disediakan di setiap item.
* **Pengajuan Surat**Formulir digital memungkinkan karyawan memilih kategori surat, mengisi keterangan, dan mengirim pengajuan melalui endpoint /api/pengajuan. Validasi input dijalankan di sisi klien sebelum permintaan dikirim.
* **Riwayat Pengajuan**Halaman histori menampilkan daftar pengajuan pengguna dengan status terkini, diakses melalui endpoint /api/histori. Pengguna dapat melihat detail dan mengunduh dokumen hasil persetujuan.
* Desain UI/UX
* Mengikuti rancangan Figma dengan komponen Ionic seperti ion-header, ion-list, ion-card, dan ion-button.
* Warna dominan biru toska dan putih dipertahankan untuk konsistensi identitas visual.
* Penggunaan ikon dari **Ionicons** untuk memperkuat representasi fungsi pada tombol navigasi.
* Pengujian dan Debugging
* Proses pengujian dilakukan menggunakan **Ionic Lab** dan **Android Emulator** untuk memastikan tampilan dan fungsionalitas konsisten di berbagai ukuran layar.
* **Chrome DevTools** digunakan untuk debugging serta inspeksi *network* pada saat integrasi API.

Dengan implementasi ini, aplikasi mobile memberikan pengalaman pengguna yang mulus, responsif, dan memudahkan karyawan dalam melakukan pengajuan dan pelacakan surat kapan saja dan di mana saja.

4.5 Scope of Work Aplikasi

| No | Fitur | Deskripsi Singkat | **Status** |
| --- | --- | --- | --- |
| 1 | Autentikasi Pengguna | Login menggunakan email dan password dengan role admin dan karyawan | **Selesai** |
| 2 | Pengelolaan Template Surat | Admin dapat menambahkan, melihat, dan mengedit template surat | **Selesai** |
| 3 | Pengajuan Surat | Karyawan dapat mengajukan permohonan surat berdasarkan template yang tersedia | **Selesai** |
| 4 | Riwayat Surat | Karyawan dapat melihat surat-surat yang sudah diterbitkan dan statusnya | **Selesai** |
| 5 | Status Pengajuan | Menampilkan daftar pengajuan yang sedang menunggu proses | **Selesai** |
| 6 | Kelola User & Karyawan | Admin dapat menambah/mengelola data pengguna dan profil karyawan | **Selesai** |
| 7 | Approval Surat | Admin menyetujui atau menolak pengajuan surat dari karyawan | **Selesai** |
| 8 | Notifikasi | Memberi tahu user jika surat sudah disetujui atau ditolak | **Belum Dibuat** |
| 9 | Reset Password via Email | Fitur lupa password dengan pengiriman token ke email | **Belum Dibuat** |
| 10 | Dashboard Statistik Admin | Grafik/angka ringkasan pengajuan, surat selesai, dan pengguna aktif | **Belum Dibuat** |

4.6 Cuplikan Kode Program

Berisi potongan kode program dari bagian penting aplikasi. Tampilkan kode yang relevan dan mudah dipahami, seperti fungsi utama, controller, atau query yang digunakan dalam fitur tertentu.

### 4.6.1 Frontend (Angular / Ionic)

* Login page – login.page.ts

login() {

this.auth.login({ email: this.email, password: this.password }).subscribe({

next: (res: any) => {

const token = res.access\_token;

localStorage.setItem('token', token);

this.auth.getProfile().subscribe((profile: any) => {

const role = (profile.role || '').toLowerCase();

if (role !== 'karyawan') {

this.presentToast('Akun bukan karyawan');

return;

}

localStorage.setItem('user', JSON.stringify(profile));

this.router.navigate(['/home']);

});

},

error: async () => {

this.handleLoginError();

}

});

}

private async handleLoginError() {

this.triggerShakeAnimation();

this.email = '';

this.password = '';

await this.presentToast('Email atau password salah!');

}

Penjelasan singkat mengenai kode tersebut :

Potongan kode berikut menangani proses login pada sisi frontend. Ketika pengguna mengisi email dan password lalu menekan tombol login, fungsi login() akan mengirim data ke backend melalui AuthService. Jika berhasil, token disimpan di localStorage, lalu pengguna akan diarahkan ke halaman utama (/home). Jika gagal login, input akan dikosongkan dan notifikasi kesalahan ditampilkan.

* Login page – login.page,html

<ion-content [fullscreen]="true" class="login-background">

<div class="login-container">

<ion-card class="login-card" [class.shake]="isLoginError">

<ion-card-header class="login-card-header">

<ion-avatar class="login-icon">

<img src="assets/icon/user-icon.svg" alt="User Icon" />

</ion-avatar>

<ion-card-title class="login-title">Selamat Datang</ion-card-title>

<p class="login-subtitle">Silakan masuk ke akun Anda</p>

</ion-card-header>

<ion-card-content>

<form (ngSubmit)="login()">

<ion-item class="input-field" lines="inset">

<ion-label position="floating">Email</ion-label>

<ion-input type="email" [(ngModel)]="email" name="email" required autocomplete="username"></ion-input>

</ion-item>

<ion-item class="input-field" lines="inset">

<ion-label position="floating">Password</ion-label>

<ion-input type="password" [(ngModel)]="password" name="password" required autocomplete="current-password"></ion-input>

</ion-item>

<div class="forgot-password">

<a href="#">Lupa password?</a>

</div>

<ion-button expand="block" type="submit" class="login-button" shape="round" color="primary">

<ion-icon name="log-in-outline" slot="start"></ion-icon>

Login

</ion-button>

</form>

</ion-card-content>

</ion-card>

</div>

</ion-content>

Penjelasan singkat mengenai kode tersebut:

Kode HTML ini membentuk tampilan antarmuka pengguna untuk halaman login. Pengguna diminta mengisi email dan password. Tampilan dirancang agar responsif dan user-friendly menggunakan komponen Ionic seperti ion-card, ion-input, dan ion-button. Fitur tambahan seperti animasi kesalahan login (shake) dan tautan lupa password juga disediakan.

* pengajuan-surat.page.ts

login() {

this.auth.login({ email: this.email, password: this.password }).subscribe({

next: (res: any) => {

const token = res.access\_token;

localStorage.setItem('token', token);

this.auth.getProfile().subscribe((profile: any) => {

const role = (profile.role || '').toLowerCase();

if (role !== 'karyawan') {

this.presentToast('Akun bukan karyawan');

return;

}

localStorage.setItem('user', JSON.stringify(profile));

this.router.navigate(['/home']);

});

},

error: async () => {

this.handleLoginError();

}

});

}

Penjelasan singkat mengenai kode tersebut:

Potongan kode berikut menangani proses login pada sisi frontend. Ketika pengguna mengisi email dan password lalu menekan tombol login, fungsi login() akan mengirim data ke backend melalui AuthService. Jika berhasil, token disimpan di localStorage, lalu pengguna akan diarahkan ke halaman utama (/home). Jika gagal login, input akan dikosongkan dan notifikasi kesalahan ditampilkan.

* pengajuan-surat.page.html

<ion-card class="login-card" [class.shake]="isLoginError">

<ion-card-header>

<ion-avatar><img src="assets/icon/user-icon.svg" /></ion-avatar>

<ion-card-title>Selamat Datang</ion-card-title>

<p>Silakan masuk ke akun Anda</p>

</ion-card-header>

<ion-card-content>

<form (ngSubmit)="login()">

<ion-item lines="inset">

<ion-label position="floating">Email</ion-label>

<ion-input type="email" [(ngModel)]="email" name="email" required></ion-input>

</ion-item>

<ion-item lines="inset">

<ion-label position="floating">Password</ion-label>

<ion-input type="password" [(ngModel)]="password" name="password" required></ion-input>

</ion-item>

<div class="forgot-password"><a href="#">Lupa password?</a></div>

<ion-button expand="block" type="submit" shape="round" color="primary">

<ion-icon name="log-in-outline" slot="start"></ion-icon>

Login

</ion-button>

</form>

</ion-card-content>

</ion-card>

Penjelasan singkat mengenai kode tersebut:

Kode HTML ini membentuk tampilan antarmuka pengguna untuk halaman login. Pengguna diminta mengisi email dan password. Tampilan dirancang agar responsif dan user-friendly menggunakan komponen Ionic seperti ion-card, ion-input, dan ion-button. Fitur tambahan seperti animasi kesalahan login (shake) dan tautan lupa password juga disediakan.

* status-pengajuan.page.ts

ngOnInit() {

this.api.getStatusPengajuan().subscribe({

next: res => {

this.daftarStatus = res.map(item => ({

nomor: `#${item.id}`,

jenis: item.template?.judul\_template || 'Tidak diketahui',

status: item.status === 'menunggu' ? 'Menunggu Validasi' : item.status,

warna: this.getWarna(item.status),

}));

}

});

}

getWarna(status: string): string {

switch (status.toLowerCase()) {

case 'menunggu': return 'medium';

case 'disetujui': return 'success';

case 'ditolak': return 'danger';

default: return 'light';

}

}

Penjelasan singkat mengenai kode tersebut:

* Fungsi ngOnInit() dipanggil saat halaman dimuat. Ia melakukan request ke backend menggunakan api.getStatusPengajuan() untuk mengambil data status surat.
* Data kemudian dipetakan (menggunakan map) ke bentuk yang lebih mudah ditampilkan di tampilan HTML, seperti nomor, jenis, status, dan warna badge (warna).
* Fungsi getWarna(status) digunakan untuk memberikan warna yang berbeda pada setiap status:
* menunggu = abu-abu (medium)
* disetujui = hijau (success)
* ditolak = merah (danger)
* status-pengajuan.page.html

<ion-content>

<div \*ngFor="let surat of daftarStatus" class="status-card">

<div class="status-info">

<div class="status-nomor">

<strong>Nomor Surat</strong><br /> {{ surat.nomor }}

</div>

<div class="status-jenis">{{ surat.jenis }}</div>

<ion-badge [color]="surat.warna">{{ surat.status }}</ion-badge>

</div>

</div>

</ion-content>

Penjelasan singkat mengenai kode tersebut:

* Menggunakan struktur \*ngFor untuk menampilkan daftar status surat secara dinamis.
* Informasi yang ditampilkan mencakup:
* Nomor surat (#id)
* Jenis surat (diambil dari judul template)
* Status (dengan badge berwarna untuk membedakan secara visual)
* riwayat-surat.page.ts

ngOnInit() {

this.api.getRiwayatSurat().subscribe({

next: (data) => {

this.riwayatSurat = data.map((item: any) => ({

nomor: item.nomor\_surat || 'SURAT/TIDAK/VALID',

jenis: item.template?.judul\_template || 'Tidak diketahui',

tanggal: item.tanggal\_diterbitkan

? new Date(item.tanggal\_diterbitkan).toLocaleDateString('id-ID', {

day: '2-digit', month: 'long', year: 'numeric'

}) : 'Tanggal tidak tersedia',

status: item.status || 'tidak diketahui',

warna:

item.status === 'selesai' ? 'success' :

item.status === 'ditolak' ? 'danger' :

item.status === 'menunggu' ? 'warning' : 'medium',

fileUrl: item.template?.file\_path

? `https://dagloktim.my.id/download-template/${encodeURIComponent(item.template.file\_path.split('/').pop())}`

: null

}));

}

});

}

Penjelasan singkat mengenai kode tersebut:

Menampilkan seluruh riwayat surat yang telah diterbitkan, termasuk tanggal, jenis, dan statusnya.

* riwayat-surat.page.html

<ion-content>

<ion-searchbar [(ngModel)]="keyword" placeholder="Cari nomor, jenis, atau status..."></ion-searchbar>

<ion-card \*ngFor="let item of filteredSurat">

<ion-card-content>

<h5>{{ item.jenis }}</h5>

<p>{{ item.nomor }} • {{ item.tanggal }}</p>

<ion-badge [color]="item.warna">{{ item.status }}</ion-badge>

<ion-button \*ngIf="item.fileUrl" (click)="lihatDokumen(item.fileUrl)">

DOWNLOAD TEMPLATE

</ion-button>

<p \*ngIf="!item.fileUrl">File belum tersedia</p>

</ion-card-content>

</ion-card>

</ion-content>

Penjelasan singkat mengenai kode tersebut:

Menampilkan daftar riwayat surat dengan fitur pencarian, status badge, dan tombol unduh jika file tersedia.

* token.interceptor.ts

@Injectable()

export class TokenInterceptor implements HttpInterceptor {

constructor(private authService: AuthService) {}

intercept(req: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {

const token = this.authService.getToken();

if (token) {

const authReq = req.clone({

setHeaders: {

Authorization: `Bearer ${token}`

}

});

return next.handle(authReq);

}

return next.handle(req);

}

}

Penjelasan singkat mengenai kode tersebut:

Kode ini adalah HTTP interceptor yang secara otomatis menambahkan token autentikasi (Bearer Token) ke setiap permintaan HTTP. Fungsinya adalah memastikan semua request ke backend memiliki otorisasi yang valid sesuai token yang didapat saat login.

* auth.guard.ts

export const authGuard: CanActivateFn = () => {

const authService = inject(AuthService);

const router = inject(Router);

const isAuthenticated = authService.isAuthenticated();

const user = JSON.parse(localStorage.getItem('user') || 'null');

if (!isAuthenticated || !user || user.role?.toLowerCase() !== 'karyawan') {

router.navigate(['/login']);

return false;

}

return true;

};

Penjelasan singkat mengenai kode tersebut:

Kode ini merupakan guard yang mencegah pengguna tidak sah mengakses halaman tertentu. Jika pengguna belum login atau bukan dengan peran "karyawan", maka akan diarahkan kembali ke halaman login. Guard ini digunakan untuk mengamankan route pada aplikasi.

### 4.6.2 Backend (Laravel)

* AuthController

public function apiLogin(Request $request)

{

$credentials = $request->only('email', 'password');

if (!$token = JWTAuth::attempt($credentials)) {

return response()->json(['error' => 'Unauthorized'], 401);

}

return response()->json([

'access\_token' => $token,

'token\_type' => 'bearer',

'expires\_in' => auth('api')->factory()->getTTL() \* 60

]);

}

Penjelasan:

Fungsi apiLogin digunakan untuk menangani proses login melalui API. Fungsi ini:

* Menerima email dan password dari request.
* Memverifikasi kredensial menggunakan JWTAuth::attempt().
* Jika berhasil, mengembalikan JWT token, jenis token, dan masa berlaku (expires\_in) dalam bentuk JSON.
* Jika gagal, mengembalikan respons "Unauthorized" dengan status 401.
* PengajuanSuratApiController

public function store(Request $request)

{

$request->validate([

'user\_id' => 'required|exists:users,id',

'template\_id' => 'required|exists:template\_surat,id',

'tanggal\_pengajuan' => 'required|date',

'status' => 'required|in:menunggu,disetujui,ditolak',

'keterangan' => 'nullable|string',

'tanggal\_berlaku' => 'required|date',

'tanggal\_berakhir' => 'required|date'

]);

PengajuanSurat::create([

'user\_id' => $request->user\_id,

'template\_id' => $request->template\_id,

'tanggal\_pengajuan' => $request->tanggal\_pengajuan,

'status' => $request->status,

'keterangan' => $request->keterangan,

'tanggal\_berlaku' => $request->tanggal\_berlaku,

'tanggal\_berakhir' => $request->tanggal\_berakhir,

]);

return response()->json(['message' => 'Pengajuan surat berhasil disimpan'], 201);

}

public function getStatusPengajuan()

{

$user = Auth::guard('api')->user();

$pengajuan = PengajuanSurat::with('template')

->where('user\_id', $user->id)

->orderBy('created\_at', 'desc')

->get();

return response()->json($pengajuan);

}

public function getRiwayat()

{

$user = Auth::guard('api')->user();

$riwayat = PengajuanSurat::with('template')

->where('user\_id', $user->id)

->where('status', '!=', 'menunggu')

->orderBy('created\_at', 'desc')

->get();

return response()->json($riwayat);

}

Penjelasan:

* Fungsi store() digunakan untuk menyimpan data pengajuan surat dari pengguna. Data divalidasi terlebih dahulu sebelum disimpan ke database.
* Fungsi getStatusPengajuan() mengembalikan seluruh data pengajuan dari pengguna yang sedang login, berguna untuk fitur *Status Pengajuan*.
* Fungsi getRiwayat() menampilkan data pengajuan yang sudah diproses (tidak berstatus “menunggu”), digunakan untuk fitur *Riwayat Surat*.
* Semua endpoint dijalankan dalam konteks pengguna yang telah diautentikasi menggunakan JWT (auth:api middleware).
* HistoriSuratApiController

public function getRiwayatUser()

{

$user = Auth::guard('api')->user();

$riwayat = HistoriSurat::where('user\_id', $user->id)

->with('template') // ambil relasi data template surat

->orderBy('created\_at', 'desc')

->get();

return response()->json($riwayat);

}

Penjelasan:

* Fungsi getRiwayatUser() digunakan untuk mengambil **riwayat surat** milik user yang sedang login.
* Data yang ditampilkan akan mencakup relasi ke tabel template, untuk menampilkan judul atau informasi template surat terkait.
* Pengambilan data diurutkan berdasarkan waktu (created\_at) dari yang terbaru ke yang terlama.
* Endpoint ini dilindungi dengan autentikasi menggunakan JWT melalui Auth::guard('api').
* routes/api.php

use Illuminate\Support\Facades\Route;

use App\Http\Controllers\AuthController;

use App\Http\Controllers\api\TemplateSuratApiController;

use App\Http\Controllers\api\PengajuanSuratApiController;

use App\Http\Controllers\api\RiwayatSuratApiController;

// Login & Logout

Route::post('/login', [AuthController::class, 'apiLogin']);

Route::middleware('auth:api')->post('/logout', function () {

auth()->logout();

return response()->json(['message' => 'Successfully logged out']);

});

// User profile

Route::middleware('auth:api')->get('/me', function (Request $request) {

return $request->user();

});

// Template surat

Route::get('/template-surat', [TemplateSuratApiController::class, 'index']);

// Pengajuan surat

Route::post('/pengajuan-surat', [PengajuanSuratApiController::class, 'store']);

Route::middleware('auth:api')->get('/pengajuan/status', [PengajuanSuratApiController::class, 'getStatusPengajuan']);

Route::middleware('auth:api')->get('/pengajuan/riwayat', [RiwayatSuratApiController::class, 'getRiwayatUser']);

Penjelasan:

* **/login**: Endpoint untuk login user dan mengembalikan token JWT.
* **/logout**: Endpoint logout untuk menghapus token dan keluar dari sistem.
* **/me**: Mengambil data user yang sedang login.
* **/template-surat**: Menampilkan daftar template surat yang tersedia.
* **/pengajuan-surat**: Menyimpan pengajuan surat baru ke database.
* **/pengajuan/status**: Menampilkan status pengajuan surat user.
* **/pengajuan/riwayat**: Menampilkan riwayat surat milik user yang sudah diproses.

Semua endpoint yang memerlukan autentikasi dilindungi oleh middleware auth:api.

# BAB V PENGUJIAN SISTEM

5.1 Metode Pengujian (Blackbox)

Pengujian blackbox dilakukan dengan cara memberikan input tertentu pada fitur aplikasi dan mengamati output yang dihasilkan. Hasil output akan dibandingkan dengan output yang diharapkan untuk mengetahui apakah fitur berjalan dengan benar. Pengujian ini tidak melibatkan pemeriksaan terhadap kode program, melainkan hanya pada aspek fungsionalitas sistem dari sisi pengguna.

5.2 Tabel Hasil Pengujian Fitur

| No | Nama Fitur | Skenario Pengujian | Input | Output yang Diharapkan | Hasil |
| --- | --- | --- | --- | --- | --- |
| 1 | Login | Login dengan username dan password valid | Username: admin  Password: 123456 | Pengguna diarahkan ke halaman beranda | Sesuai |
| 2 | Login | Login dengan data tidak valid | Username: adminPassword: salah | Muncul notifikasi "Login gagal" | Sesuai |
| 3 | Pengajuan Surat | Pengajuan surat baru dengan semua data terisi lengkap | Jenis surat, tanggal, isi surat | Data berhasil dikirim dan masuk ke daftar pengajuan | Sesuai |
| 4 | Pengajuan Surat | Pengajuan surat dengan data kosong | - | Muncul notifikasi kesalahan (validasi form) | Sesuai |
| 5 | Riwayat Surat | Menampilkan riwayat surat yang pernah diajukan | - | Daftar surat yang pernah diajukan muncul di layar | Sesuai |
| 6 | Status Pengajuan | Mengecek status dari surat yang diajukan | - | Status ditampilkan dengan benar (menunggu, disetujui) | Sesuai |
| 7 | Template Surat | Menampilkan daftar template surat | - | Semua template surat muncul lengkap dan dapat dipilih | Sesuai |
| 8 | Logout | Menekan tombol logout | - | Pengguna keluar dan diarahkan ke halaman login | Sesuai |

# BAB VI PENERAPAN DAN UJI COBA

6.1 Uji Coba ke Pengguna / Stakeholder

Uji coba dilakukan dengan memberikan akses sistem kepada sejumlah pengguna internal, yaitu anggota tim dan beberapa rekan mahasiswa lain yang disimulasikan sebagai karyawan dan admin. Metode uji coba dilakukan secara langsung melalui demonstrasi aplikasi mobile dan web, disertai pengisian survei evaluasi sederhana.

Skenario uji meliputi:

* Login ke sistem
* Mengajukan surat
* Menyetujui surat sebagai admin
* Melihat riwayat pengajuan

6.2 Dokumentasi Instalasi / Manual Pengguna

* Langkah Instalasi Backend:
* Pastikan XAMPP/Laragon dan Composer telah terpasang.
* Clone repositori Laravel melalui GitHub.
* Jalankan perintah composer install.
* Buat file .env dan sesuaikan konfigurasi database.
* Jalankan php artisan migrate dan php artisan serve.
* Langkah Instalasi Aplikasi Mobile:
* Pastikan Node.js dan Ionic CLI telah terpasang.
* Clone proyek Ionic dari repositori GitHub.
* Jalankan npm install.
* Gunakan perintah ionic serve untuk menjalankan aplikasi di browser.
* Untuk pengujian mobile, gunakan ionic capacitor run android.
* Panduan Pengguna:
* Login menggunakan email dan kata sandi yang diberikan.
* Pilih fitur sesuai kebutuhan: ajukan surat, lihat status, atau unduh template.
* Admin dapat mengakses fitur validasi dan laporan melalui halaman dashboard.

6.3 Evaluasi Hasil Uji Coba (Feedback Pengguna)

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Pengguna** | **Masukan / Feedback** | **Tanggapan Tim Developer** |
| 1 | Bapak Gio | UI sederhana dan mudah digunakan, namun fitur cetak belum ada | Akan ditambahkan pada versi berikutnya |
| 2 | Mbak Anisa | Respon aplikasi cepat, riwayat surat sangat membantu | Terima kasih, akan terus dipertahankan |
| 3 | Pak Fajar | Dashboard informatif, tetapi kurang filter per kategori | Fitur filter lanjutan sedang dikembangkan |

6.4 Rencana Pengembangan Lebih Lanjut

Berdasarkan hasil evaluasi dan keterbatasan saat pengembangan awal, berikut beberapa rencana pengembangan berikutnya:

* Menambahkan fitur ekspor dan cetak surat dalam format PDF.
* Integrasi dengan email atau WhatsApp untuk notifikasi otomatis.
* Penambahan fitur pencarian riwayat surat berdasarkan NIK atau kategori.
* Pengembangan versi iOS dari aplikasi mobile.
* Pengamanan tambahan seperti autentikasi dua faktor.

# BAB VII PENUTUP

## 7.1 Kesimpulan

Setelah melalui serangkaian tahapan yang meliputi analisis kebutuhan, perancangan sistem, implementasi, hingga pengujian menyeluruh, akhirnya aplikasi *Sistem File Archive Perusahaan* berhasil dikembangkan dan diimplementasikan secara nyata. Aplikasi ini lahir dari keprihatinan terhadap proses surat-menyurat di perusahaan yang selama ini masih bersifat manual, rumit, lambat, dan rentan kesalahan. Kami hadir membawa solusi digital yang tidak hanya menyederhanakan proses tersebut, tetapi juga menghadirkan pengalaman baru bagi karyawan dan HRD dalam mengelola dokumen perusahaan secara efisien dan terstruktur.

Kini, seorang karyawan tidak perlu lagi mencetak formulir, mengisi surat dengan tangan, atau mengantri di meja HRD. Cukup dari genggaman tangan melalui aplikasi mobile, mereka dapat mengakses template resmi perusahaan, mengajukan permohonan surat, memantau status pengajuan, bahkan mengunduh surat resmi yang sudah disetujui – semua dilakukan secara **real-time, aman, dan terdokumentasi**. Sementara bagi HRD, sistem ini mempermudah proses verifikasi, pelacakan, dan rekap surat dengan penomoran otomatis yang rapi dan akurat.

Seluruh fitur utama yang dirancang dalam perencanaan telah berhasil diimplementasikan dan diuji. Dengan demikian, **tujuan proyek dinyatakan tercapai** secara keseluruhan, baik dari sisi fungsionalitas, manfaat, maupun kemudahan penggunaan.

## 7.2 Saran

Meskipun sistem ini telah berjalan dengan baik, kami menyadari bahwa pengembangan teknologi adalah proses berkelanjutan. Oleh karena itu, kami memberikan beberapa saran untuk pengembangan di masa mendatang:

1. **Penambahan fitur notifikasi real-time**, agar pengguna segera mendapatkan informasi saat surat mereka disetujui atau ditolak tanpa harus membuka aplikasi secara berkala.
2. **Fitur reset password via email** perlu diaktifkan sepenuhnya untuk mendukung keamanan dan kenyamanan pengguna saat lupa kredensial login.
3. **Peningkatan tampilan antarmuka (UI/UX)** agar tampilan lebih modern, dinamis, dan ramah pengguna di berbagai ukuran layar perangkat.
4. **Dashboard statistik yang lebih informatif** baik untuk admin maupun karyawan, agar mereka dapat memantau jumlah pengajuan, status terkini, serta tren aktivitas secara visual.
5. **Integrasi penyimpanan cloud** seperti Google Drive atau AWS S3 agar file arsip lebih aman, mudah diakses, dan mendukung skalabilitas di masa depan.
6. **Ekspansi sistem untuk skala perusahaan lebih besar** dengan tambahan role pengguna baru seperti supervisor atau manajer divisi yang dapat turut memverifikasi atau memantau surat dari timnya.

Dengan pengembangan ini, kami yakin sistem dapat menjangkau lebih luas, mendukung digitalisasi perusahaan secara menyeluruh, dan menjadi solusi praktis dalam manajemen dokumen di berbagai sektor.

# LAMPIRAN

- Kode Program  
- Desain UI  
- Tabel Pengujian  
- Feedback Pengguna  
- dll

**Catatan:**

Dalam penyusunan laporan ini, mahasiswa diwajibkan menggunakan kaidah Bahasa Indonesia yang baik dan benar, mengacu pada Pedoman Umum Ejaan Bahasa Indonesia (PUEBI), struktur penulisan ilmiah, dan kelengkapan unsur kebahasaan seperti penggunaan kata baku, kalimat efektif, serta koherensi antar paragraf.