LAPORAN KERJA PRAKTEK

RANCANG BANGUN BACKEND SISTEM PENDAFTARAN MAGANG DAN PENELITIAN BERBASIS WEB DINAS KESEHATAN KOTA SURABAYA

Oleh:

Nalendra Putra Wicaksana

1462300034

PROGRAM SARJANA

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK ELEKTRO

DAN INFORMATIKA CERDAS

UNIVERSITAS 17 AGUSTUS 1945 SURABAYA

2026

LEMBAR PENGESAHAN

RANCANG BANGUN BACKEND SISTEM PENDAFTARAN MAGANG DAN PENELITIAN BERBASIS WEB DINAS KESEHATAN KOTA SURABAYA

Sebagai salah satu syarat untuk melaksanakan Kerja Praktek

Oleh:

Nalendra Putra Wicaksana

1462300034

Surabaya, 06 Januari 2026

Koordinator KP,	Dosen Pembimbing
Anton Breva Yunanda, S.T., M.MT	Intan Dzikria, S.Kom., M.IM., Ph.D
NPP. 20450020554	NPP. 20460160701

Mengetahui,

Ka, Program Studi Teknik Informatika

Puteri Noraisya Primandari, S.ST., M.IM.

NPP. 20460170736

KATAPENGANTAR

Alhamdulillahirabbil 'aalamiin, puji syukur kepada Allah SWT, karena dengan rahmat, hidayah, dan karunia-Nya, penulis dapat menyelesaikan proposal kerja praktek ini dengan judul "Rancang Bangun Backend Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya", sebagai syarat untuk melaksanakan kerja praktek pada Program Studi Teknik Informatika, Fakultas Teknik, Universitas 17 Agustus 1945 Surabaya.

Penyusunan proposal ini bertujuan untuk memberikan gambaran mengenai rencana kegiatan kerja praktek yang akan dilaksanakan di Dinas Kesehatan Kota Surabaya. Melalui kegiatan ini, penulis berharap dapat memperoleh pengalaman dan wawasan yang bermanfaat, baik dalam bidang akademik maupun dunia kerja, khususnya dalam pengembangan backend sistem informasi berbasis web

Penulis menyadari bahwa tanpa adanya dukungan dan bimbingan dari berbagai pihak, pelaksanaan kerja praktek hingga penyusunan laporan ini tidak akan terselesaikan dengan baik. Oleh karena itu, penulis ingin menyampaikaan ucapa terima kasih yang sebesar – besarnya kepada :

1. Puteri Noraisya Primandari S.ST., M.IM.selaku Ketua Program Studi Teknik Informatika yang telah memberikan dukungan dan fasilitas dalam pelaksanaan kerja praktek.
2. Anton Breva Yunanda S.T., M.MT selaku Dosen Koordinator Kerja Pratek yang telah memberikan arahan dan kebijaksanaan dalam pelaksanaan kerja praktek.
3. Intan Dzikria, S.Kom., M.IM., Ph.D selaku Dosen Pembimbing yang telah memberikan bimbingan, saran, dan masukan yang sangat bermanfaat dalam penyusunan proposal ini.
4. dr. Reyner Meilaksana Sumbung selaku Pembina Tingkat I yang telah menerima untuk melakukan kegiatan kerja praktek pada Dinas Kesehatan Kota Surabaya.
5. Seluruh pimpinan dan staf Dinas Kesehatan Kota Surabaya yang telah memberikan pengalaman dan ilmunya selama pelaksanaan kerja praktek ini.

6. Serta untuk kedua orang tua, kakak dan keponakan yang selalu memberikan do'a dan dukunganya untuk sampai pada titik ini.

Penulis menyadari bahwa penulisan laporan ini masih jauh dari kata sempurna. Oleh karena itu, setiap kritik dan saran dapat membantu penulis untuk dapat menjadi lebih baik dikemudian hari. Diharapkan dari penulisan laporan ini, pembaca dapat mengerti, memahami, serta menambah referesni dalam membangun jaringan komputer baru.

Surabaya, Januari 2026

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi dan komunikasi yang pesat telah mendorong berbagai sektor, termasuk instansi pemerintahan, untuk mengadopsi sistem informasi yang andal dan efisien. Dalam lingkup pelayanan administrasi, kebutuhan terhadap sistem yang mampu menunjang kegiatan operasional harian, pengawasan keamanan pengelolaan data, serta transparansi informasi menjadi sangat penting. Sistem informasi yang terstruktur dengan baik akan meningkatkan efisiensi, produktivitas, dan kualitas layanan kepada masyarakat maupun pihak eksternal yang membutuhkan akses.

Namun, tidak sedikit instansi yang masih mengandalkan proses administrasi manual, seperti penggunaan dokumen fisik, pencatatan terpisah, serta belum adanya integrasi sistem. Hal ini dapat menimbulkan berbagai hambatan, seperti lamanya proses verifikasi data, penumpukan berkas, risiko kehilangan dokumen, hingga keterbatasan akses informasi bagi pendaftar.

Untuk menjawab tantangan tersebut, dibutuhkan perancangan dan implementasi sistem pendaftaran yang memperhatikan aspek kebutuhan operasional, efisiensi, keamanan data, serta kemudahan akses. Dalam konteks ini, sistem berbasis web menjadi solusi yang tepat karena menawarkan fleksibilitas, ketersediaan akses secara real-time, serta kemampuan integrasi dengan layanan lain. Backend sistem memegang peranan penting dalam hal pengaturan logika bisnis, validasi data, manajemen basis data, serta integrasi dengan antarmuka pengguna melalui sistem routing dan controller.

Melalui kerja praktek ini, penulis terlibat langsung dalam proyek pembangunan backend sistem pendaftaran magang dan penelitian di bawah naungan Dinas Kesehatan Kota Surabaya. Keterlibatan tersebut mencakup analisis kebutuhan, perancangan basis data, implementasi sistem backend, hingga pengujian layanan. Diharapkan, proyek ini dapat memberikan kontribusi nyata dalam peningkatan kualitas administrasi pendaftaran magang dan penelitian, serta menjadi pengalaman aplikatif dalam penerapan ilmu di bidang pengembangan sistem informasi berbasis web.

1.2 Tujuan

Tujuan dari kerja prakter ini adalah sebagai berikut .

- Merancang dan membangun backend sistem pendaftaran magang dan penelitian berbasis web pada Dinas Kesehatan Kota Surabaya.
- Mengimplementasikan fungsi manajemen data pendaftar, verifikasi, serta status pengajuan secara terintegrasi
- Meningkatkan kemampuan komunikasi, kolaborasi, dan manajemen waktu dalam lingkungan kerja serta belajar bekerja secara mandiri dan dalam tim.
- Meningkatkan efisiensi dan akurasi dalam pengelolaan data pendaftaran

1.3 Manfaat

Manfaat dari pelaksanaan kerja praktek ini meliputi

- Mempermudah administrasi pendaftaran di Dinas Kesehatan Kota Surabaya.
- Mempermudah proses pengelolaan data pendaftar magang dan penelitian.
- Menyediakan system backend yang terintegritas dengan antarmuka pengguna.
- Meningkatkan efisiensi dan akurasi pengelohan data.

1.4 Luaran

Luaran dari kerja praktek ini adalah backend sistem pendaftaran magang dan penelitian berbasis web yang terintegrasi dengan basis data, disertai dokumentasi teknis serta laporan kerja praktek sebagai hasil akhir.

1.5 Waktu Dan Tempat Pelaksanaan

Tempat Kerja Praktek dilaksanakan di:

Tempat	: Dinas Kesehatan Kota Surabaya
Alamat	: Jl. Raya Jemursari No.197, Sidosermo, Kec. Wonocolo, Surabaya, Jawa Timur 60239
Tanggal	: 01 Agustus 2025 s.d. 31 Oktober 2025
Waktu	: 07.30 s.d. 16.00

BAB 2
GAMBARAN UMUM

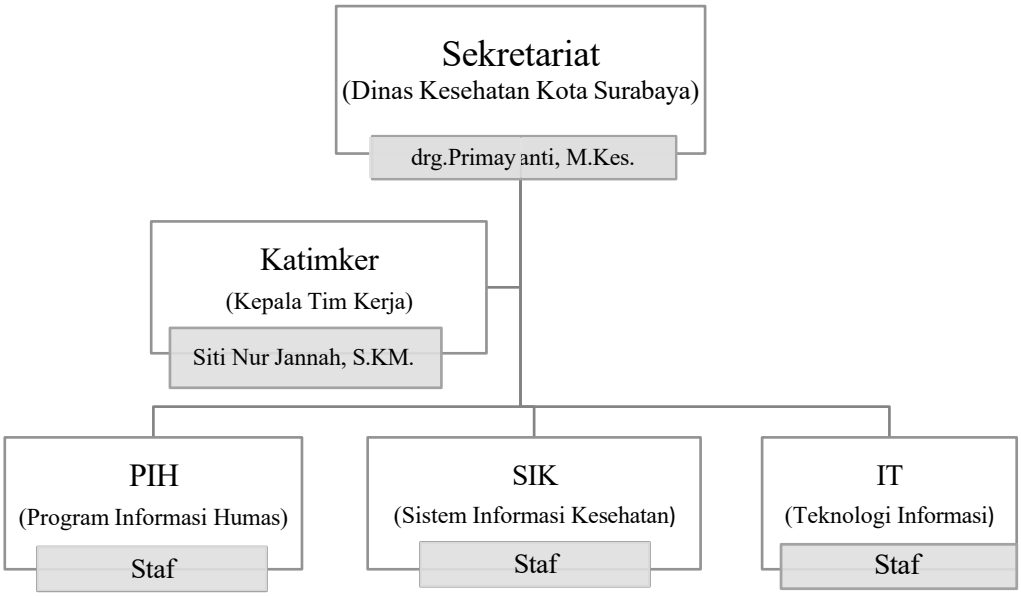
2.1 Profil

Dinas Kesehatan Kota Surabaya, sebagai unsur pelaksana utama Pemerintah Kota Surabaya di bidang kesehatan masyarakat, secara resmi berpusat di Kantor yang representatif dan strategis, yang beralamat di Jl. Jemursari No. 197, Surabaya. Kantor ini menjadi pusat koordinasi seluruh upaya kesehatan publik di kota metropolitan ini. Seluruh operasional dan mandat yang diemban oleh Dinas Kesehatan ditegaskan Sesuai dengan landasan hukum yang kuat, yakni Peraturan Walikota Surabaya Nomor 48 Tahun 2016 tentang Rincian Tugas dan Fungsi Dinas Kesehatan Kota Surabaya. Berdasarkan Perwali tersebut, Dinas mempunyai tanggung jawab dan tugas pokok untuk melaksanakan urusan pemerintahan daerah di bidang kesehatan secara menyeluruh berdasarkan azas otonomi daerah dan berperan aktif dalam melaksanakan tugas pembantuan di bidang kesehatan. Tugas ini merupakan manifestasi komitmen Pemerintah Kota Surabaya dalam peningkatan derajat dan kualitas kesehatan warga.

Untuk menjamin efektivitas dan efisiensi dalam menyelenggarakan tugas pokok dan strategis sebagaimana tersebut di atas, Dinas Kesehatan Kota Surabaya menjalankan serangkaian fungsi penting yang terstruktur, antara lain:

1. Perumusan kebijakan sesuai dengan lingkup tugasnya.
2. Pelaksanaan kebijakan sesuai dengan lingkup tugasnya.
3. Pelaksanaan evaluasi dan pelaporan sesuai dengan lingkup tugasnya.
4. Pelaksanaan Administrasi dinas sesuai dengan lingkup tugasnya.
5. Pelaksanaan tugas lain yang diberikan oleh Kepala Daerah sesuai dengan tugas dan fungsinya.

2.2 Struktur Organisasi



Gambar 2. 1 Struktur Organisasi

Dari gambar 2.1. dapat dijelaskan bahwa kegiatan pada Dinas Kesehatan Kota Surabaya ini dibawahi langsung oleh Ibu drg. Primayanti, M.Kes. selaku Sekretariat yang bertanggung jawab atas koordinasi serta pengelolaan kegiatan administrasi dan operasional. Di bawah beliau terdapat Ibu Siti Nur Jannah, S.KM. selaku Kepala Tim Kerja (Katimker) yang mengatur pelaksanaan kegiatan teknis di lapangan. Pelaksanaan kegiatan didukung oleh tiga bagian utama, yaitu Program Informasi Humas (PIH) yang berperan dalam pengelolaan informasi dan komunikasi publik, Sistem Informasi Kesehatan (SIK) yang menangani pengumpulan serta pengolahan data kesehatan, dan Teknologi Informasi (IT) yang bertanggung jawab terhadap pengelolaan infrastruktur serta dukungan teknis. Seluruh bagian tersebut bekerja sama secara terstruktur mulai dari tahap perencanaan, pelaksanaan, hingga evaluasi agar kegiatan di Dinas Kesehatan Kota Surabaya dapat berjalan dengan efektif, efisien, dan terarah.

2.3 Visi dan Misi Instansi

2.3.1. Visi

Dinas Kesehatan yang Profesional untuk Gotong Royong Menuju Kota Dunia yang Maju, Humanis dan Berkelanjutan

2.3.2. Misi

Membangun SDM Unggul, Sehat Jasmani Rohani, Produktif dan Berkarakter, Melalui Peningkatan Akses dan Kualitas Pelayanan Kesehatan, Pendidikan dan Kebutuhan Dasar Lainnya

.

2.4. Aplikasi Berbasis Web Pendaftaran Dan Penelitian Dinas Kesehatan Kota Surabaya

Sistem pendaftaran magang dan penelitian berbasis web merupakan sebuah sistem informasi yang dirancang untuk mengelola proses pendaftaran secara digital, mulai dari pengisian formulir, pengunggahan dokumen, verifikasi data, hingga pemberitahuan status pengajuan kepada peserta. Sistem ini hadir untuk menggantikan proses manual yang selama ini masih menggunakan berkas fisik dan verifikasi langsung di kantor, yang dinilai kurang efisien dari segi waktu dan sumber daya.

.

Menurut Rahayu dan Nurdin (2021), "Penerapan sistem informasi berbasis web dalam layanan administrasi publik dapat meningkatkan efisiensi, transparansi, dan akurasi data karena seluruh proses dilakukan secara terpusat dan terdokumentasi secara elektronik". Dengan sistem digital, proses pendaftaran dapat dilakukan kapan saja dan di mana saja, tanpa harus datang langsung ke kantor instansi.

Dalam konteks pelaksanaan magang dan penelitian di instansi pemerintahan, Dinas Kesehatan Kota Surabaya membutuhkan sistem yang mampu menampung data calon peserta, melakukan verifikasi berkas secara daring, serta memberikan informasi status pengajuan secara real-time. Sistem ini tidak hanya berfungsi sebagai sarana pendaftaran, tetapi juga sebagai alat manajerial untuk membantu pihak dinas dalam memantau jumlah pendaftar,

memeriksa kelengkapan dokumen, serta mengatur jadwal pelaksanaan kegiatan magang dan penelitian.

Fungsi utama dari sistem pendaftaran magang dan penelitian berbasis web ini meliputi:

- Formulir pendaftaran online, yang memungkinkan calon peserta mengisi data pribadi, riwayat pendidikan, dan tujuan magang/penelitian secara langsung melalui website.
- Fitur unggah dokumen, seperti surat pengantar, CV, dan proposal penelitian, yang tersimpan secara otomatis dalam basis data.
- Verifikasi data dan dokumen oleh admin Dinas Kesehatan melalui panel kontrol.
- Notifikasi status pengajuan, yang dikirimkan kepada pendaftar melalui email secara otomatis.
- Dashboard admin yang menampilkan statistik pendaftaran dan mempermudah proses monitoring.

Pengembangan sistem berbasis web dengan model arsitektur *client–server* memberikan fleksibilitas tinggi dan kemudahan integrasi antara sisi pengguna (frontend) dan sisi pengelola data (backend). Dalam sistem ini, pengguna hanya berinteraksi dengan antarmuka web, sedangkan seluruh proses logika bisnis dan manajemen data dijalankan di server.

Pada sisi teknis, pengembangan sistem ini menggunakan framework Laravel, salah satu framework PHP yang banyak digunakan dalam pengembangan aplikasi web modern. Menurut Doroodchi dan Dastgheib (2008), "Laravel mendukung pola Model–View–Controller (MVC) yang memisahkan logika program, tampilan, dan pengolahan data sehingga sistem menjadi lebih terstruktur, mudah dikembangkan, dan aman". Beberapa fitur bawaan Laravel seperti routing, middleware, authentication, dan Eloquent ORM digunakan untuk "mempercepat proses pengembangan dan menjaga kualitas sistem" (Doroodchi & Dastgheib, 2008). Dengan diterapkannya sistem pendaftaran magang dan penelitian berbasis web ini, Dinas Kesehatan Kota Surabaya memperoleh berbagai manfaat, antara lain:

- Proses pendaftaran menjadi lebih cepat, efisien, dan minim kesalahan input data.
- Pendaftar dapat memantau status pengajuan secara transparan dan real-time.
- Admin dapat mengelola dan memverifikasi data dengan mudah, tanpa harus melakukan pencatatan manual.
- Seluruh data tersimpan secara aman dalam basis data terpusat, sehingga meminimalkan risiko kehilangan atau duplikasi berkas.

Secara keseluruhan, sistem ini berperan penting dalam mendukung digitalisasi proses administrasi di lingkungan Dinas Kesehatan, sekaligus menjadi sarana pembelajaran nyata bagi mahasiswa dalam menerapkan ilmu pengembangan sistem informasi berbasis web di dunia kerja.

.

BAB 3
PELAKSANAAN KERJA PRAKTEK

3.1. Kegiatan Survei Lapangan

Kegiatan survei lapangan pada kerja praktek ini dilakukan di Dinas Kesehatan Kota Surabaya yang berlokasi di Jl. Raya Jemursari No. 197, Sidosermo, Kecamatan Wonocolo, Surabaya, Jawa Timur 60239. Pelaksanaan kerja praktek berlangsung mulai tanggal 1 Agustus 2025 hingga 31 Oktober 2025 dan dilaksanakan Work From Office (WFO) dengan waktu kegiatan mengikuti jam kerja instansi, yaitu pukul 07.30 s.d. 16.00 WIB..

Pada awal pelaksanaan kerja praktek, mahasiswa diperkenalkan dengan struktur organisasi serta bagian-bagian yang berperan dalam kegiatan administrasi dan pelayanan publik di lingkungan Dinas Kesehatan Kota Surabaya. Dari hasil pengenalan tersebut, mahasiswa memperoleh gambaran umum mengenai proses administrasi pendaftaran magang dan penelitian yang selama ini masih dilakukan secara manual. Proses tersebut melibatkan pengumpulan berkas fisik, verifikasi data secara langsung, serta komunikasi melalui surat menyurat atau email yang tidak terintegrasi dalam satu sistem.

Selama kegiatan survei, mahasiswa mendapatkan arahan langsung dari pihak pembimbing lapangan di Dinas Kesehatan untuk melakukan analisis terhadap proses pendaftaran yang sedang berjalan, serta mengidentifikasi kendala yang sering muncul, seperti keterlambatan verifikasi, kesalahan pencatatan data, dan kesulitan dalam pelacakan status pengajuan peserta. Dari hasil survei ini, disimpulkan bahwa dibutuhkan sebuah sistem pendaftaran berbasis web yang mampu memfasilitasi proses pendaftaran, verifikasi, dan pelaporan data secara lebih efisien.

Tahapan awal kegiatan kerja praktek dimulai dengan penyusunan rancangan sistem (system design) yang meliputi perancangan alur proses, struktur basis data, serta perancangan antarmuka pengguna. Perancangan dilakukan berdasarkan hasil observasi dan diskusi bersama staf administrasi Dinas Kesehatan agar sistem yang dikembangkan sesuai dengan kebutuhan instansi.

Setelah rancangan sistem disetujui, mahasiswa kemudian melanjutkan ke tahap implementasi backend dan pengujian fungsi sistem pendaftaran magang dan penelitian menggunakan framework Laravel.

Kegiatan survei lapangan ini memberikan pemahaman yang mendalam mengenai kondisi dan kebutuhan nyata di lapangan, sekaligus menjadi dasar dalam pengembangan sistem pendaftaran magang dan penelitian berbasis web yang diharapkan dapat membantu Dinas Kesehatan Kota Surabaya dalam meningkatkan efisiensi, transparansi, dan kualitas pelayanan administrasi.

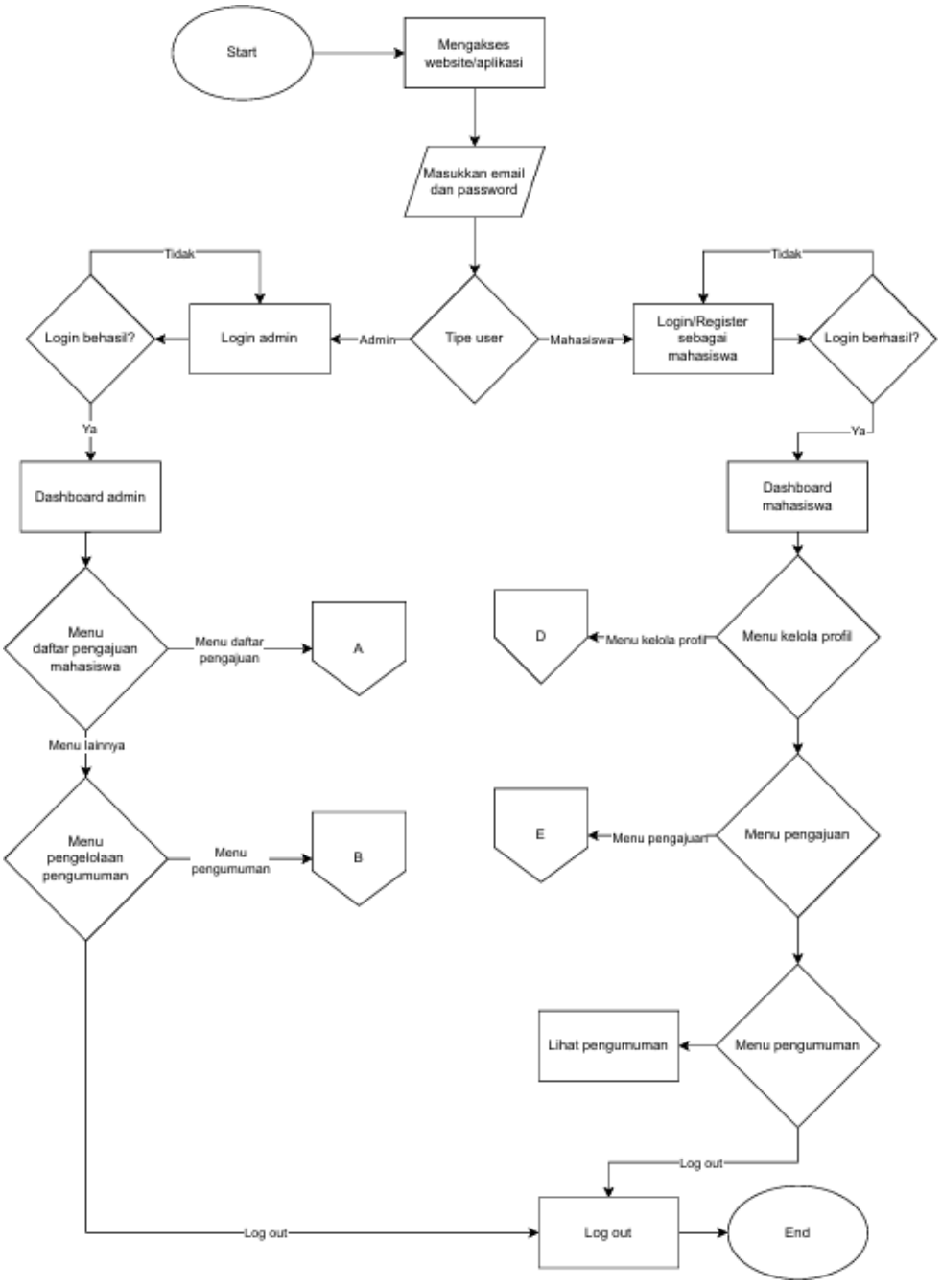
3.2. Proses Bisnis Dan Interaksi Pengguna Dengan Sistem

3.2.1. Flowchart

Flowchart adalah diagram alur yang digunakan untuk menggambarkan langkah-langkah atau algoritma dalam suatu sistem secara visual dengan simbol-simbol standar. Diagram ini membantu analis dan programmer memahami alur logika sistem secara sistematis, mulai dari awal, proses, pengambilan keputusan, hingga akhir.

Flowchart memudahkan komunikasi antara tim teknis dan non-teknis karena alur kerja sistem dapat dipahami dengan jelas. Menurut Rosaly and Prasetyo (2020), "selain sebagai dokumentasi, flowchart juga menjadi pedoman dalam pengembangan backend untuk memastikan proses, logika bisnis, dan alur data berjalan sesuai dengan perancangan awal".

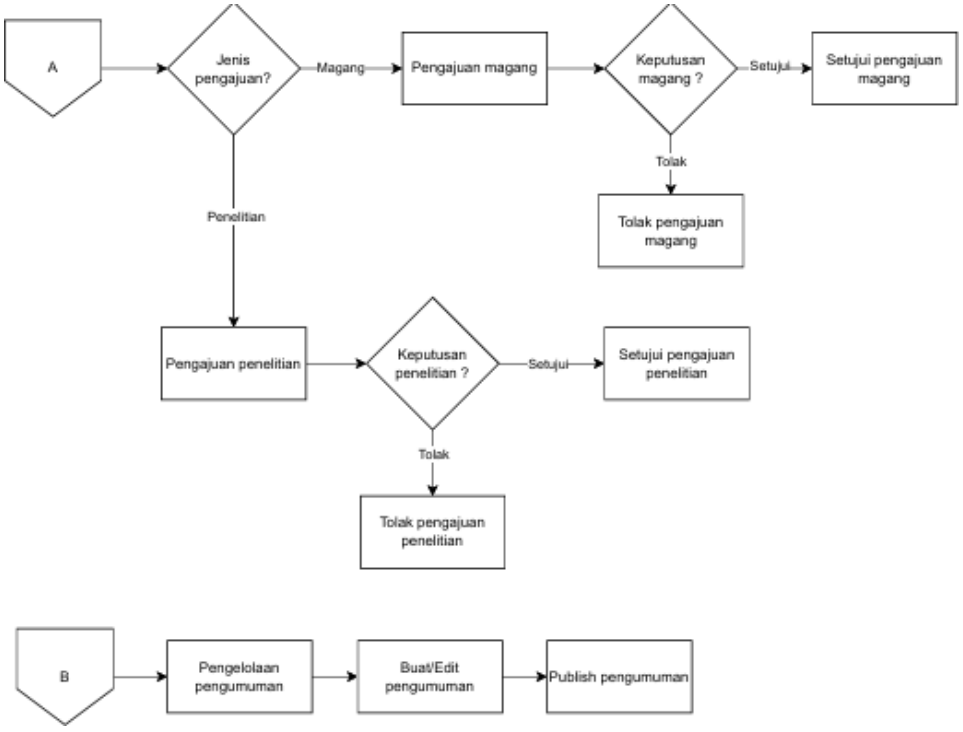
Berikut merupakan flowchart proses sistem yang menggambarkan alur kerja dari fitur-fitur yang telah dikembangkan dalam Sistem Pendaftaran Magang dan Penelitian Berbasis Web di Dinas Kesehatan Kota Surabaya**.**



Gambar 3. 1 Flowchart Sistem Pendaftaran Magang dan Penelitian

Pada gambar 3.1 di atas merupakan proses penggunaan sistem secara keseluruhan yang menggambarkan alur kerja baik dari sisi mahasiswa maupun admin Dinas Kesehatan. Flowchart dimulai dengan simbol oval bertuliskan "Start", yang menandakan awal dari proses. Setelah itu, pengguna diarahkan untuk melakukan aksi pertama yaitu mengakses website atau aplikasi Sistem Pendaftaran Magang dan Penelitian. Tahap ini merupakan pintu masuk utama agar pengguna dapat menggunakan layanan yang tersedia di dalam sistem.

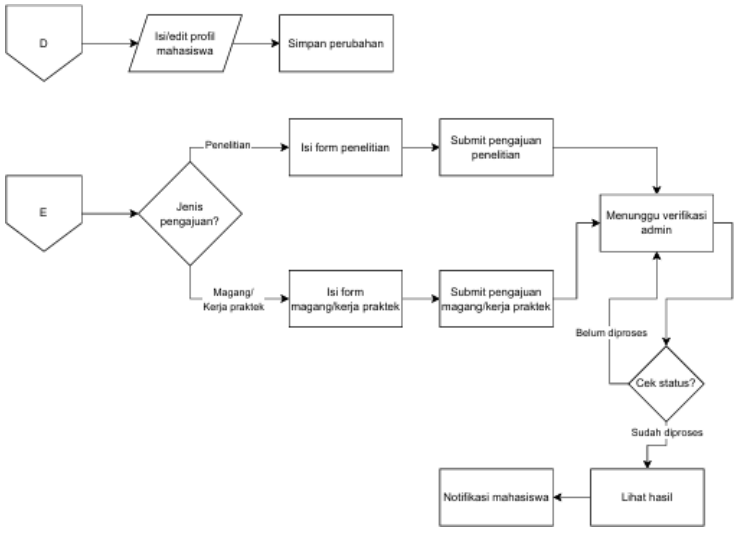
Selanjutnya, sistem akan mendeteksi tipe user yang mengakses melalui proses identifikasi pengguna. Pada tahap ini terdapat dua pilihan tipe user, yaitu mahasiswa dan admin. Percabangan ini menjadi poin penting karena menentukan alur proses selanjutnya yang akan dijalani oleh masing-masing pengguna sesuai dengan peran dan kewenangannya dalam sistem.



Gambar 3. 2 Flowchart Admin

Pada Gambar 3.2 ditunjukkan alur kerja admin dalam Sistem Pendaftaran Magang dan Penelitian. Proses dimulai ketika admin mengakses sistem dan diarahkan ke halaman login. Admin diminta untuk memasukkan email dan password sebagai kredensial. Sistem akan melakukan verifikasi terhadap data yang dimasukkan. Jika login gagal, admin akan dikembalikan ke halaman login untuk mencoba kembali. Jika login berhasil, admin akan diarahkan ke dashboard admin sebagai pusat pengelolaan sistem.

Melalui dashboard, admin dapat mengakses menu daftar pengajuan mahasiswa yang berisi pengajuan magang dan penelitian. Admin melakukan pemeriksaan kelengkapan dokumen serta kesesuaian persyaratan, kemudian menentukan keputusan apakah pengajuan disetujui atau ditolak. Sistem akan memperbarui status pengajuan sesuai keputusan admin dan mengirimkan notifikasi kepada mahasiswa. Proses yang sama juga berlaku untuk pengajuan penelitian.
Selain mengelola pengajuan, admin dapat mengelola pengumuman dengan membuat atau mengedit informasi penting, serta mengelola profil admin. Setelah seluruh proses selesai, admin dapat melakukan logout untuk keluar dari sistem. Flowchart diakhiri dengan simbol *End* yang menandakan bahwa seluruh alur proses telah selesai.



Gambar 3. 3 Flowchart Pengguna

Pada Gambar 3.3 ditunjukkan alur penggunaan sistem dari sisi mahasiswa. Ketika mahasiswa mengakses sistem, mereka diarahkan ke halaman login atau registrasi. Mahasiswa yang belum memiliki akun dapat melakukan registrasi terlebih dahulu, sedangkan yang sudah terdaftar dapat langsung login menggunakan email dan password. Sistem akan memverifikasi data login, jika salah mahasiswa dikembalikan ke halaman login, dan jika benar mahasiswa berhasil masuk ke dashboard.

Setelah login, mahasiswa masuk ke dashboard mahasiswa yang menyediakan beberapa menu utama. Melalui menu pengajuan, mahasiswa dapat mengajukan permohonan magang atau penelitian. Jika memilih pengajuan penelitian, mahasiswa harus mengisi form penelitian berupa judul, tujuan, metodologi, serta mengunggah dokumen pendukung, kemudian melakukan submit. Pengajuan akan masuk ke sistem untuk diverifikasi oleh admin, dan mahasiswa dapat memantau statusnya melalui menu daftar pengajuan. Mahasiswa juga akan menerima notifikasi hasil verifikasi apakah pengajuan disetujui atau ditolak.

Proses pengajuan magang atau kerja praktik dilakukan dengan cara yang serupa. Mahasiswa mengisi form magang yang mencakup periode, bidang yang diminati, tujuan magang, serta dokumen persyaratan, kemudian melakukan submit. Status pengajuan dapat dipantau melalui menu daftar pengajuan hingga admin memberikan keputusan dan sistem mengirimkan notifikasi hasilnya.

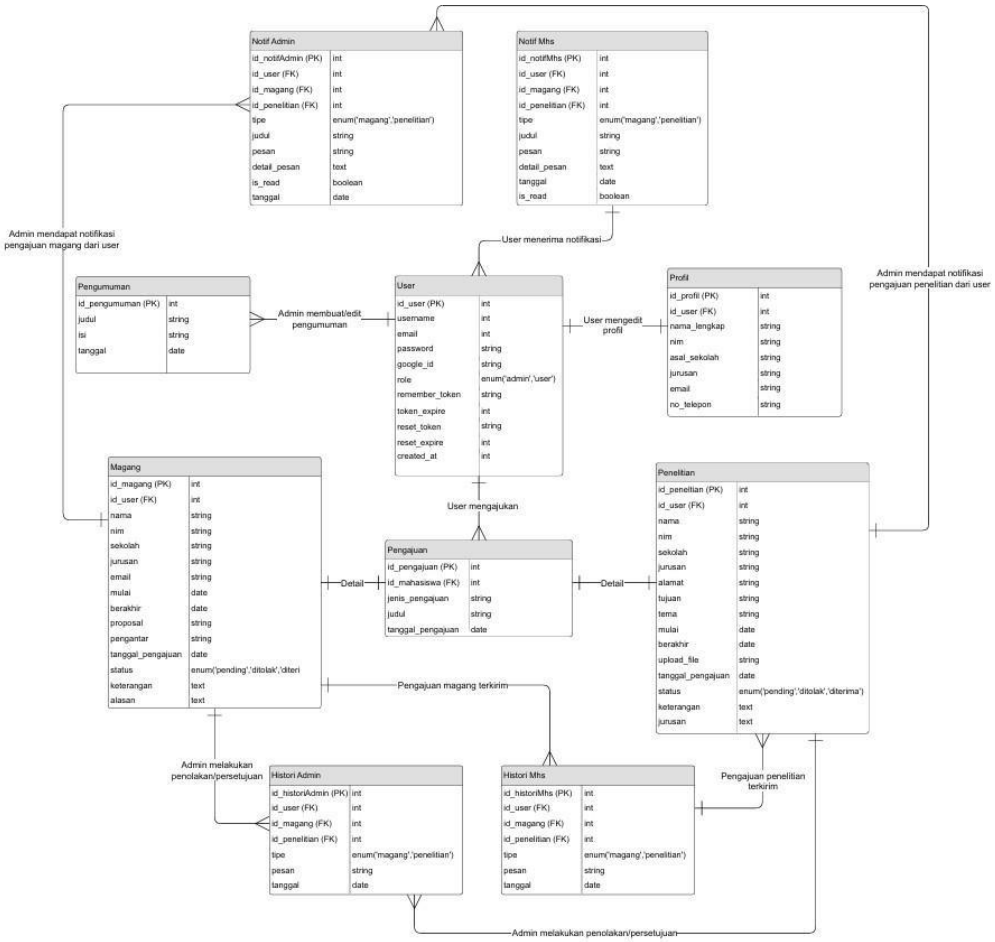
Selain pengajuan, mahasiswa juga dapat mengakses menu daftar pengajuan untuk melihat riwayat pengajuan, menu pengumuman untuk memperoleh informasi dari Dinas Kesehatan, serta menu kelola profil untuk mengubah data pribadi dan akademik. Setelah selesai menggunakan sistem, mahasiswa dapat melakukan logout untuk keluar dari dashboard dengan aman.

.

3.2.2. ER Diagram

Entity Relationship Diagram (ERD) merupakan "model data yang dikembangkan berdasarkan objek dan digunakan untuk menggambarkan hubungan antar data dalam basis data secara logis" (Putra, Ferdinandus and Bayu, 2019). ERD membantu merancang struktur database dengan cara yang mudah dipahami, bahkan oleh pengguna yang belum berpengalaman secara teknis. Model ini memvisualisasikan entitas, atribut, dan relasi yang saling terhubung di dalam sistem. Entitas digambarkan sebagai objek nyata seperti orang atau benda, atribut berfungsi sebagai informasi penjelas untuk entitas, sedangkan relasi menggambarkan hubungan antar entitas tersebut. Menurut Putra, Ferdinandus and Bayu (2019), "penggunaan ERD sangat membantu dalam proses analisis dan perancangan sistem karena mampu menunjukkan macam data yang dibutuhkan serta bagaimana relasi antar data tersebut terhubung". Dengan adanya ERD, perancang sistem dapat membentuk struktur database yang efisien, terorganisir, dan sesuai kebutuhan aplikasi yang dikembangkan.

Berikut gambar 3.4 *Entity Relationship Diagram* Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya.



Gambar 3. 4 *Entity Relationship Diagram*

Berdasarkan gambar 3.4 diagram ERD (*Entity Relationship Diagram*) dari database sistem pendaftaran magang dan penelitian, dapat dijelaskan bahwa struktur ini dirancang untuk mendukung sistem pendaftaran secara menyeluruh, mulai dari pengelolaan data pengguna, profil mahasiswa, pengajuan magang dan penelitian, hingga sistem notifikasi dan histori aktivitas. Database ini juga menyediakan sistem pengumuman dan manajemen peran pengguna, yang penting untuk membedakan hak akses antara admin dan mahasiswa sebagai user.

1. Tabel User dan Autentikasi.

   Tabel User menyimpan informasi dasar pengguna sistem seperti id_user sebagai primary key, username, email, password, google_id untuk autentikasi melalui Google, role yang bertipe enum dengan nilai 'admin' atau 'user' untuk membedakan tipe pengguna, remember_token dan token_expire untuk manajemen sesi login, reset_token dan reset_expire untuk fitur reset password, serta created_at untuk mencatat waktu pembuatan akun. Tabel ini menjadi pusat dari seluruh sistem karena setiap aktivitas dalam aplikasi terkait dengan user yang melakukan tindakan tersebut. Pengguna dalam sistem ini memiliki hubungan one-to-one dengan tabel Profil untuk menyimpan informasi detail mahasiswa, serta hubungan one-to-many dengan tabel Penelitian dan Magang karena satu user dapat mengajukan beberapa permohonan penelitian atau magang. Selain itu, tabel User juga memiliki relasi one-to-many dengan tabel Notif Mhs, Notif Admin, Histori Mhs, dan Histori Admin untuk mencatat semua notifikasi dan aktivitas yang dilakukan oleh pengguna.

2. Tabel Profil Mahasiswa.

   Tabel Profil berisi informasi lengkap tentang mahasiswa yang terdaftar dalam sistem. Tabel ini memiliki id_profil sebagai primary key dan id_user sebagai foreign key yang menghubungkan dengan tabel User. Atribut yang tersimpan meliputi nama_lengkap, nim (Nomor Induk Mahasiswa), asal_sekolah atau universitas, no_telepon untuk keperluan komunikasi, jurusan atau program studi, dan email. Relasi antara User dan Profil bersifat one-to-one, yang artinya setiap user mahasiswa memiliki satu profil lengkap. Tabel ini memungkinkan mahasiswa untuk mengelola dan memperbarui informasi pribadi mereka melalui fitur kelola profil yang tersedia di dashboard

3. Tabel Penelitian.

Tabel Penelitian menyimpan seluruh data pengajuan penelitian yang diajukan oleh mahasiswa. Tabel ini memiliki id_penelitian sebagai primary key dan id_user sebagai foreign key yang menunjuk ke tabel User. Atribut yang disimpan meliputi nama pemohon, nim, sekolah atau universitas asal, jurusan, alamat, tujuan penelitian, tema penelitian yang akan diteliti, tanggal mulai dan berakhir penelitian, upload_file untuk menyimpan path file proposal atau dokumen pendukung, tanggal_pengajuan untuk mencatat kapan pengajuan disubmit, status dengan nilai enum 'pending', 'ditolak', atau 'diterima' untuk menunjukkan kondisi pengajuan, serta keterangan yang berisi catatan atau feedback dari admin. Relasi antara User dan Penelitian bersifat many-to-one, yang berarti satu mahasiswa dapat mengajukan beberapa permohonan penelitian dalam waktu yang berbeda. Tabel Penelitian juga memiliki relasi one-to-many dengan tabel Notif Mhs, Notif Admin, Histori Mhs, dan Histori Admin untuk mencatat notifikasi dan histori terkait pengajuan penelitian tersebut.

4. Tabel Magang.

Tabel Magang memiliki struktur yang serupa dengan tabel Penelitian namun dirancang khusus untuk pengajuan magang atau kerja praktek. Tabel ini memiliki id_magang sebagai primary key dan id_user sebagai foreign key. Atribut yang tersimpan meliputi nama pemohon, nim, sekolah, jurusan, email, tanggal mulai dan berakhir magang, proposal untuk menyimpan file proposal magang, pengantar untuk file surat pengantar dari institusi, tanggal_pengajuan, status dengan nilai enum 'pending', 'ditolak', atau 'diterima', keterangan dari admin, dan alasan yang dapat diisi ketika pengajuan ditolak.

Seperti tabel Penelitian, relasi antara User dan Magang juga bersifat many-to-one karena satu mahasiswa dapat mengajukan beberapa permohonan magang. Tabel Magang juga memiliki relasi one-to-many dengan tabel Notif Mhs, Notif Admin, Histori Mhs, dan Histori Admin untuk keperluan notifikasi dan pencatatan histori aktivitas.

5. Tabel Pengumuman

Tabel Pengumuman dikelola oleh admin untuk menyampaikan informasi penting kepada seluruh pengguna sistem. Tabel ini memiliki struktur sederhana dengan id_pengumuman sebagai primary key, judul pengumuman, isi yang berisi konten pengumuman, dan tanggal publikasi. Admin dapat membuat, mengedit, dan mempublikasikan pengumuman melalui dashboard admin. Pengumuman ini dapat berisi informasi tentang jadwal penerimaan magang atau penelitian, perubahan kebijakan, pengumuman hasil seleksi, atau informasi penting lainnya yang perlu diketahui oleh mahasiswa. Tabel ini tidak memiliki relasi foreign key dengan tabel lain karena bersifat independen dan dapat diakses oleh semua pengguna.

6. Tabel Notif Mhs (Notifikasi Mahasiswa)

Tabel Notif Mhs berfungsi untuk menyimpan notifikasi yang diterima oleh mahasiswa. Tabel ini memiliki id_notifMhs sebagai primary key, id_user sebagai foreign key yang menunjuk ke mahasiswa penerima notifikasi, id_magang dan id_penelitian sebagai foreign key yang menunjuk ke pengajuan terkait, tipe dengan nilai enum 'magang' atau 'penelitian' untuk membedakan jenis notifikasi, judul notifikasi, pesan singkat, detail_pesan yang berisi informasi lengkap, is_read dengan tipe boolean untuk menandai apakah notifikasi sudah dibaca, dan tanggal pengiriman notifikasi.

Relasi tabel ini bersifat many-to-one dengan tabel User, many-to-one dengan tabel Magang, dan many-to-one dengan tabel Penelitian. Sistem notifikasi ini membantu mahasiswa mendapatkan informasi real-time ketika pengajuan mereka diproses, disetujui, atau ditolak oleh admin.

7. Tabel Notif Admin (Notifikasi Admin)

Tabel Notif Admin memiliki struktur yang sama dengan Notif Mhs namun ditujukan untuk admin. Tabel ini berisi id_notifAdmin sebagai primary key, id_user untuk admin penerima, id_magang dan id_penelitian untuk pengajuan yang terkait, tipe notifikasi, judul, pesan, detail_pesan, is_read, dan tanggal. Relasi tabel ini bersifat many-to-one dengan tabel User, many-to-one dengan tabel Magang, dan many-to-one dengan tabel Penelitian. Admin akan menerima notifikasi setiap kali ada pengajuan baru dari mahasiswa, baik itu pengajuan magang maupun penelitian, sehingga admin dapat segera melakukan verifikasi dan memberikan keputusan.

8. Tabel Histori Mhs (Histori Mahasiswa)

Tabel Histori Mhs berfungsi untuk menyimpan riwayat aktivitas mahasiswa dalam sistem. Tabel ini memiliki id_historiMhs sebagai primary key, id_user untuk mengidentifikasi mahasiswa, id_magang dan id_penelitian untuk mencatat pengajuan mana yang terkait dengan histori tersebut, tipe aktivitas dengan nilai enum 'magang' atau 'penelitian', pesan yang menjelaskan aktivitas yang dilakukan, dan tanggal aktivitas. Relasi tabel ini bersifat many-to-one dengan tabel User, many-to-one dengan tabel Magang, dan many-to-one dengan tabel Penelitian. Histori ini berguna untuk tracking aktivitas mahasiswa seperti kapan pengajuan dibuat, diubah, atau dibatalkan, sehingga memberikan transparansi dan dokumentasi yang lengkap.

9. Tabel Histori Admin (Histori Admin)

Tabel Histori Admin memiliki struktur yang serupa dengan Histori Mhs namun mencatat aktivitas yang dilakukan oleh admin. Tabel ini berisi id_historiAdmin sebagai primary key, id_user untuk admin yang melakukan aktivitas, id_magang dan id_penelitian untuk pengajuan terkait, tipe aktivitas, pesan yang menjelaskan tindakan admin seperti menyetujui atau menolak pengajuan, dan tanggal aktivitas. Relasi tabel ini bersifat many-to-one dengan tabel User, many-to-one dengan tabel Magang, dan many-to-one dengan tabel Penelitian. Histori admin penting untuk audit trail dan memastikan akuntabilitas dalam proses verifikasi pengajuan.

10. Tabel Pengajuan.

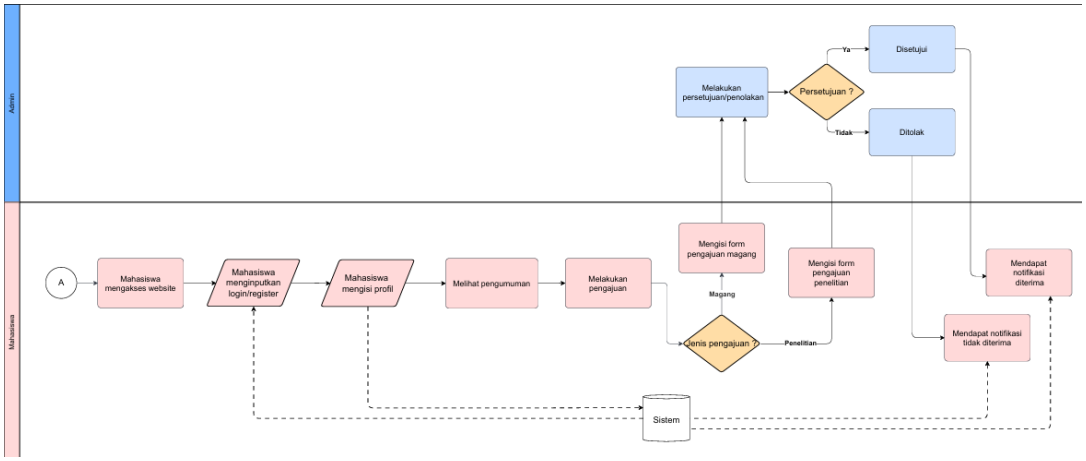
Terakhir, tabel pengajuan berfungsi sebagai tabel ringkasan yang menyimpan informasi umum tentang semua pengajuan dalam sistem. Tabel ini memiliki id_pengajuan sebagai primary key, id_mahasiswa sebagai foreign key yang menunjuk ke user pemohon dengan relasi many-to-one, jenis_pengajuan yang menunjukkan apakah pengajuan tersebut untuk magang atau penelitian, judul pengajuan, dan tanggal_pengajuan. Tabel ini memudahkan admin untuk melihat daftar semua pengajuan dalam satu tempat sebelum melihat detail lebih lanjut di tabel Penelitian atau Magang.

3.2.3. Proses Bisnis

Proses bisnis merupakan serangkaian aktivitas atau tugas yang terstruktur dan saling berkaitan untuk mencapai tujuan tertentu dalam suatu organisasi. Dalam konteks sistem informasi, proses bisnis menggambarkan alur kerja yang melibatkan interaksi antara pengguna, sistem, dan stakeholder lainnya.

Diagram proses bisnis membantu memvisualisasikan bagaimana setiap entitas berinteraksi satu sama lain, bagaimana data mengalir dalam sistem, dan bagaimana setiap proses berkontribusi terhadap pencapaian tujuan organisasi. Dengan adanya dokumentasi proses bisnis yang jelas, pengembang sistem dapat memahami kebutuhan fungsional secara menyeluruh dan memastikan bahwa sistem yang dibangun sesuai dengan alur kerja yang diharapkan.

Berikut merupakan diagram proses bisnis yang menggambarkan alur kerja dari Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya.

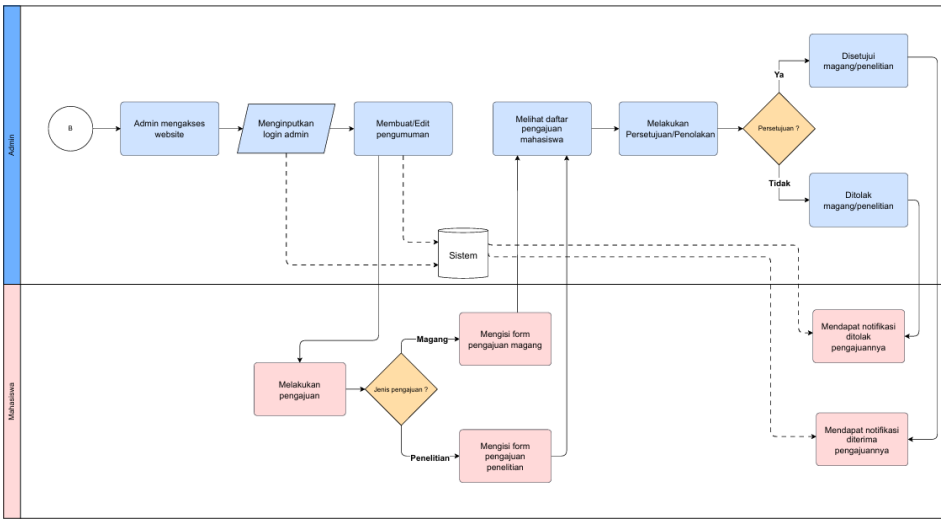


Gambar 3. 5 Proses Bisnis User

Pada Gambar 3.5 ditunjukkan alur proses bisnis dari sisi mahasiswa dalam menggunakan Sistem Pendaftaran Magang dan Penelitian. Proses dimulai saat mahasiswa mengakses website, kemudian melakukan login atau registrasi. Setelah berhasil masuk, mahasiswa melengkapi data profil sebagai identitas dalam sistem.

Mahasiswa selanjutnya mengajukan permohonan dengan memilih jenis pengajuan, yaitu magang atau penelitian, serta mengisi form dan mengunggah dokumen persyaratan. Setelah pengajuan disubmit, status akan berubah menjadi *pending* dan menunggu verifikasi admin. Admin kemudian memberikan keputusan diterima atau ditolak. Mahasiswa akan menerima notifikasi hasil melalui dashboard dan email, disertai alasan jika pengajuan ditolak.

Selain pengajuan, mahasiswa juga dapat mengakses fitur pengumuman untuk memperoleh informasi terbaru terkait jadwal, hasil seleksi, dan kebijakan magang maupun penelitian di Dinas Kesehatan.



Gambar 3. 6 Proses Bisnis Admin

Pada Gambar 3.6 ditunjukkan alur proses bisnis dari sisi admin dalam mengelola Sistem Pendaftaran Magang dan Penelitian. Proses diawali dengan admin melakukan login untuk mengakses dashboard. Melalui dashboard, admin dapat melihat daftar seluruh pengajuan mahasiswa, baik magang maupun penelitian, lengkap dengan informasi ringkas dan detail pengajuan yang dapat diperiksa lebih lanjut.

Admin melakukan verifikasi dokumen dan kelayakan pengajuan sesuai jenisnya. Pada pengajuan magang, admin memeriksa kelengkapan berkas, kesesuaian periode, unit tujuan, serta kapasitas pelaksanaan. Sementara pada pengajuan penelitian, admin mengevaluasi proposal,

relevansi tema, metodologi, dan aspek keamanan data. Setelah evaluasi, admin menentukan keputusan untuk menyetujui atau menolak pengajuan. Jika disetujui, status akan diperbarui dan mahasiswa menerima notifikasi penerimaan. Jika ditolak, status diubah menjadi ditolak disertai alasan, dan mahasiswa menerima notifikasi penolakan.

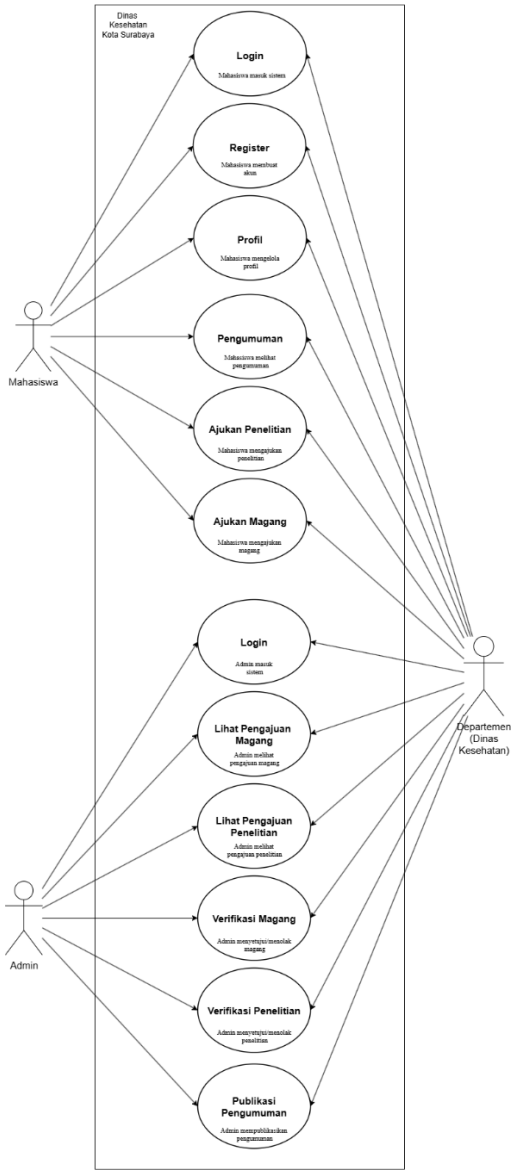
Selain mengelola pengajuan, admin juga bertugas membuat dan mengedit pengumuman yang berisi informasi penting seperti jadwal, persyaratan, dan hasil seleksi. Pengumuman yang dipublikasikan dapat diakses oleh seluruh mahasiswa, sehingga komunikasi antara Dinas Kesehatan dan mahasiswa dapat berlangsung secara efektif dan efisien.

3.2.4. Use Case Diagram

Use Case Diagram merupakan diagram yang menggambarkan interaksi antara aktor (pengguna) dengan fungsi-fungsi yang tersedia dalam sistem. Diagram ini digunakan untuk menunjukkan siapa saja yang terlibat dalam sistem dan apa saja aktivitas yang dapat mereka lakukan.

Menurut Dennis, Wixom and Tegarden (2015), Use Case Diagram sangat membantu dalam tahap analisis kebutuhan karena "memudahkan pengembang memahami kebutuhan fungsional, cakupan sistem, serta menjadi media komunikasi yang efektif antara developer dan stakeholder".

Berikut merupakan Use Case yang menggambarkan alur kerja dari Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya.



Gambar 3. 7 Use Case Diagram User dan Admin

Berdasarkan gambar 3.7 Use Case Diagram dari sistem pendaftaran magang dan penelitian Dinas Kesehatan Kota Surabaya, dapat dijelaskan bahwa sistem ini dirancang untuk melayani tiga aktor utama, yaitu Mahasiswa, Admin, dan Departemen (Dinas Kesehatan). Setiap aktor memiliki peran dan hak akses yang berbeda terhadap fungsionalitas sistem.

Mahasiswa sebagai aktor utama memiliki akses terhadap beberapa use case

- Register - Mahasiswa dapat mendaftarkan akun baru untuk mengakses sistem
- Login - Mahasiswa melakukan autentikasi untuk masuk ke dalam sistem
- Profil - Mahasiswa dapat mengelola dan memperbarui informasi profil pribadi mereka
- Pengumuman - Mahasiswa dapat melihat informasi pengumuman yang dipublikasikan oleh admin
- Form Pengajuan Penelitian - Mahasiswa mengisi dan mengajukan permohonan penelitian melalui formulir yang tersedia
- Form Pengajuan Magang - Mahasiswa mengisi dan mengajukan permohonan magang melalui formulir yang tersedia

Admin memiliki peran sebagai pengelola sistem dengan akses terhadap use case

- Login - Admin melakukan autentikasi untuk mengakses panel administrasi
- Melihat Daftar Pengajuan Magang Mahasiswa - Admin dapat menampilkan dan memantau semua pengajuan magang yang masuk
- Melihat Daftar Pengajuan Penelitian Mahasiswa - Admin dapat menampilkan dan memantau semua pengajuan penelitian yang masuk
- Menyetujui/Menolak Pengajuan Magang - Admin memberikan persetujuan atau penolakan terhadap pengajuan magang mahasiswa
- Menyetujui/Menolak Pengajuan Penelitian - Admin memberikan persetujuan atau penolakan terhadap pengajuan penelitian mahasiswa
- Mempublish Pengumuman - Admin dapat mengirimkan dan mempublikasikan pemberitahuan kepada mahasiswa

Departemen (Dinas Kesehatan Kota Surabaya) sebagai stakeholder memiliki akses penuh terhadap seluruh use case yang tersedia di sistem, mencakup semua fungsionalitas yang dapat diakses oleh Mahasiswa dan Admin. Hal ini menunjukkan bahwa Departemen memiliki peran supervisi dan monitoring terhadap keseluruhan proses pendaftaran magang dan penelitian.

3.3. Pemilihan Supporting Pengembangan Tools

3.3.1. HTML

HTML (*HyperText Markup Language*) adalah "bahasa markup standar untuk membangun struktur dasar halaman web, seperti teks, gambar, tabel, tautan, dan elemen multimedia lainnya" (Permatasari & Suhendi, 2020). Setiap elemen ditandai dengan tag yang memberi instruksi kepada browser tentang cara menampilkan konten. Dalam praktiknya, HTML digunakan bersama CSS untuk mengatur tampilan dan JavaScript untuk menambahkan interaksi sehingga halaman menjadi lebih dinamis (Permatasari & Suhendi, 2020). HTML tetap menjadi fondasi utama karena seluruh isi dan struktur halaman web dibentuk melalui bahasa ini. HTML juga memungkinkan "pembuatan tautan antarhalaman, baik dalam satu situs maupun antar situs, sehingga membentuk jaringan informasi di internet" (Permatasari & Suhendi, 2020). "Versi terbaru HTML mendukung multimedia, perangkat mobile, dan aksesibilitas yang lebih baik" sehingga penting dipahami dalam pengembangan web (Permatasari & Suhendi, 2020).

3.3.2. PHP

PHP (*HyperText PreProcessor*) adalah bahasa pemrograman skrip sisi server yang digunakan secara luas untuk membangun website dinamis. PHP berfungsi mengeksekusi logika aplikasi seperti autentikasi pengguna, pemrosesan transaksi, serta pengelolaan data melalui database menggunakan SQL, terutama MySQL dan PostgreSQL. Sebagai bahasa open source, PHP didukung komunitas besar, bersifat skalabel, dan terus berkembang hingga versi terbaru seperti PHP 8 yang lebih efisien. Dalam pengembangan modern, PHP banyak digunakan melalui framework seperti Laravel dan Symfony yang menerapkan arsitektur MVC, sehingga kode lebih terstruktur, mudah dipelihara, dan aman untuk membangun aplikasi *web professional*.

3.3.3. MySQL

MySQL adalah "sistem manajemen basis data relasional (RDBMS) open source yang banyak digunakan dalam pengembangan aplikasi web skala kecil hingga enterprise" (Suli and Nirsal, 2023). MySQL memungkinkan pengguna merancang, mengelola, dan memanipulasi database secara efisien menggunakan bahasa SQL (Structured Query Language). "MySQL mendukung fitur multiuser, multithreading, serta integrasi yang fleksibel dengan berbagai bahasa pemrograman backend, khususnya PHP" (Suli and Nirsal, 2023). Karena bersifat open source, MySQL dapat digunakan secara gratis dan didukung oleh berbagai sistem operasi. Dengan kinerja yang stabil dan keandalan pengelolaan data, "MySQL menjadi pilihan utama dalam pengembangan sistem informasi berbasis web" (Suli and Nirsal, 2023).

3.3.4. Laravel

Laravel adalah framework web open-source berbasis PHP yang populer untuk mempermudah dan mempercepat pengembangan aplikasi web secara terstruktur dengan arsitektur MVC (Model-View-Controller) (Doroodchi and Dastgheib, 2008). Laravel menyediakan berbagai fitur penting seperti routing, manajemen sesi, autentikasi pengguna, serta Eloquent ORM untuk pengelolaan database dengan sintaks yang lebih sederhana.

Dengan dukungan fitur seperti caching, queue, dan testing, serta ekosistem yang kuat, "Laravel mampu meningkatkan efisiensi pengembang dan digunakan secara luas untuk membangun aplikasi web skala kecil hingga besar dengan performa tinggi" (Doroodchi and Dastgheib, 2008).

3.3.5. Visual Studio Code

Visual Studio Code adalah "editor teks lintas platform yang dikembangkan oleh Microsoft, dirancang untuk ringan, cepat, dan kaya fitur" (Kurniawan & Romzi, 2022). Editor ini mendukung berbagai sistem operasi seperti Windows, macOS, dan Linux, serta memiliki dukungan bawaan untuk beberapa bahasa pemrograman. Kekuatan utamanya terletak pada extensions marketplace yang menyediakan ribuan ekstensi untuk mendukung berbagai bahasa, framework, dan kebutuhan pengembangan.

Visual Studio Code dilengkapi fitur seperti IntelliSense, syntax highlighting, terminal bawaan, debugger, serta integrasi langsung dengan Git untuk pengelolaan versi kode. Dengan antarmuka yang intuitif, fleksibilitas tinggi, dan kemampuan kustomisasi yang luas, "Visual Studio Code menjadi salah satu editor kode paling populer dan banyak digunakan untuk meningkatkan produktivitas pengembang" (Kurniawan & Romzi, 2022).

3.3.6. Bootstrap

Bootstrap merupakan framework front-end open-source yang populer untuk membangun antarmuka web modern, khususnya admin panel dan dashboard backend. Framework ini menyediakan berbagai komponen siap pakai seperti tabel, form, modal, tombol, navigasi, dan alert yang memudahkan pengembangan tampilan sistem secara cepat dan konsisten.

Bootstrap mendukung "desain responsif dengan sistem grid berbasis mobile-first, sehingga dashboard dapat diakses dengan baik melalui berbagai perangkat seperti komputer, tablet, dan smartphone" (Suli and Nirsal, 2023). Komponen seperti pagination, badge status, dan modal sangat membantu dalam pengelolaan data dan aktivitas admin agar lebih efisien dan intuitif. Integrasi Bootstrap dengan Laravel berlangsung dengan baik melalui template Blade, di mana data dari backend dapat ditampilkan dengan tampilan yang rapi dan terstruktur. Dengan dokumentasi lengkap, performa yang baik, serta komunitas besar, "Bootstrap menjadi solusi efektif untuk membangun sistem backend yang profesional dan responsif" (Suli and Nirsal, 2023).

3.3.7. Sweetalert

SweetAlert adalah library JavaScript yang digunakan untuk menggantikan alert bawaan browser dengan dialog notifikasi dan konfirmasi yang lebih menarik, interaktif, dan dapat dikustomisasi (Suli and Nirsal, 2023). Library ini banyak digunakan dalam aplikasi web modern untuk meningkatkan user experience melalui tampilan dialog yang profesional dan informatif. Dalam sistem backend pendaftaran magang dan penelitian, SweetAlert berperan sebagai media feedback dan konfirmasi atas aksi penting yang dilakukan admin, seperti menyetujui, menolak, atau menghapus data pengajuan. SweetAlert terintegrasi dengan Laravel melalui AJAX maupun session flash message untuk menampilkan notifikasi sukses, error, maupun peringatan secara otomatis berdasarkan respon dari backend.

SweetAlert2 sebagai versi terbaru juga mendukung input form, timer, dan berbagai tipe notifikasi seperti success, error, warning, dan info. Dengan ukuran yang ringan, dokumentasi lengkap, serta kemudahan integrasi, "SweetAlert efektif meningkatkan kejelasan interaksi, keamanan aksi kritis, dan profesionalitas tampilan sistem backend" (Suli and Nirsal, 2023).

3.3.8. Xampp

XAMPP merupakan paket perangkat lunak open source yang digunakan sebagai server lokal untuk mendukung pengembangan dan pengujian aplikasi web. XAMPP mengintegrasikan Apache, MySQL, dan PHP dalam satu paket sehingga memudahkan proses instalasi dan konfigurasi lingkungan server secara lokal (Nugroho, 2019). Dengan dukungan lintas platform dan kemudahan penggunaan, XAMPP banyak dimanfaatkan dalam pengembangan sistem informasi berbasis web (Nugroho, 2019)

3.4. Analisis Kebutuhan

3.4.1. Kebutuhan Fungsional

Kebutuhan fungsional adalah deskripsi tentang layanan atau fungsi yang harus disediakan oleh sistem perangkat lunak. Kebutuhan ini mendefinisikan apa yang harus dilakukan sistem untuk memenuhi tujuan pengguna.

Berdasarkan analisa pengguna didapatkan spesifikasi kebutuhan fungsional pada tabel 3.4.1

Tabel 3.4. 1 Kebutuhan Fungsional

NO	KODE	KEBUTUHAN FUNGSIONAL	AKTOR
1	KF01	Dapat melakukan registrasi akun baru	Mahasiswa
2	KF02	Dapat melakukan login ke sistem	Mahasiswa, Admin
3	KF03	Dapat melakukan logout dari sistem	Mahasiswa, Admin
4	KF04	Dapat melakukan reset password melalui email	Mahasiswa
5	KF05	Dapat mengelola dan memperbarui profil mahasiswa	Mahasiswa
6	KF06	Dapat mengisi dan submit form pengajuan magang	Mahasiswa
7	KF07	Dapat mengisi dan submit form pengajuan penelitian	Mahasiswa
8	KF08	Dapat mengunggah dokumen proposal dan surat pengantar (PDF, max 10MB)	Mahasiswa
9	KF09	Dapat melihat daftar pengajuan magang dan penelitian yang telah diajukan	Mahasiswa
10	KF10	Dapat memantau status pengajuan (pending, diterima, ditolak)	Mahasiswa
11	KF11	Dapat melihat histori seluruh pengajuan yang pernah dibuat	Mahasiswa
12	KF12	Dapat menerima notifikasi real-time tentang status pengajuan	Mahasiswa

13	KF13	Dapat melihat pengumuman yang dipublikasikan oleh admin	Mahasiswa
14	KF14	Dapat melihat daftar semua pengajuan magang	Admin
15	KF15	Dapat melihat daftar semua pengajuan penelitian	Admin
16	KF16	Dapat melihat detail lengkap pengajuan magang	Admin
17	KF17	Dapat melihat detail lengkap pengajuan penelitian	Admin
18	KF18	Dapat menyetujui pengajuan magang dengan menambahkan keterangan	Admin
19	KF19	Dapat menolak pengajuan magang dengan menambahkan alasan penolakan	Admin
20	KF20	Dapat menyetujui pengajuan penelitian dengan menambahkan keterangan	Admin
21	KF21	Dapat menolak pengajuan penelitian dengan menambahkan alasan penolakan	Admin
22	KF22	Dapat melihat dan mengunduh dokumen proposal dan surat pengantar	Admin
23	KF23	Dapat membuat, mengedit, dan menghapus pengumuman	Admin
24	KF24	Dapat melihat notifikasi pengajuan baru dari mahasiswa	Admin
25	KF25	Dapat menandai notifikasi sebagai sudah dibaca	Admin
26	KF26	Dapat menghapus notifikasi (single dan bulk delete)	Admin
27	KF27	Dapat melihat histori aktivitas admin dalam verifikasi pengajuan	Admin
28	KF28	Dapat menghapus histori aktivitas (single dan bulk delete)	Admin
29	KF29	Sistem dapat mengirimkan email otomatis untuk reset password	Sistem
30	KF30	Sistem dapat membuat notifikasi otomatis saat status pengajuan berubah	Sistem

3.4.2. Kebutuhan Non Fungsional

Kebutuhan non fungsional adalah deskripsi tentang kriteria yang menilai operasi sistem, bukan fungsi spesifik yang disediakannya. Kebutuhan ini mencakup karakteristik kualitas sistem seperti performa, keamanan, keandalan, dan skalabilitas.

Berdasarkan analisa pengguna didapatkan spesifikasi kebutuhan fungsional pada tabel 3.4.2

Tabel 3.4. 2 Kebutuhan Non Fungsional

NO	KODE	KEBUTUHAN NON-FUNGSIONAL	KETERANGAN
1	KNF01	Keamanan password menggunakan enkripsi bcrypt	Password disimpan dalam bentuk hash untuk keamanan data
2	KNF02	Validasi format file upload (hanya PDF)	Sistem hanya menerima dokumen dalam format PDF
3	KNF03	Batas ukuran file upload maksimal 10MB	Mencegah overload storage dan mempercepat proses upload
4	KNF04	Session-based authentication untuk kontrol akses	Menggunakan session untuk memvalidasi user yang login
5	KNF05	Role-based access control (admin dan mahasiswa)	Pembatasan akses fitur berdasarkan peran pengguna
6	KNF06	Token-based reset password dengan masa berlaku 1 jam	Token reset password kadaluarsa setelah 1 jam untuk keamanan
7	KNF07	Remember me functionality dengan durasi 30 hari	Cookie login otomatis bertahan selama 30 hari
8	KNF08	Database menggunakan MySQL dengan struktur relasional	Menggunakan foreign key dan relasi untuk integritas data
9	KNF09	Framework backend menggunakan Laravel versi 12	Implementasi MVC pattern untuk struktur kode yang terorganisir
10	KNF10	Antarmuka responsif menggunakan Bootstrap	Tampilan dapat diakses dengan baik di berbagai perangkat
11	KNF11	Notifikasi menggunakan SweetAlert untuk user experience	Feedback visual yang menarik dan informatif
12	KNF12	Penyimpanan file menggunakan Laravel Storage	File disimpan di storage/app/public/uploads dengan nama unik
13	KNF13	Validasi Input menggunakan Laravel Validation Rules	Mencegah data invalid masuk ke database
14	KNF14	Query menggunakan Eloquent ORM dan Query Builder	Meningkatkan keamanan dengan prepared statements
15	KNF15	Pagination untuk daftar pengajuan	Meningkatkan performa dengan membatasi data per halaman
16	KNF16	Real-time notification count menggunakan AJAX polling	Badge notifikasi ter-update tanpa refresh halaman
17	KNF17	Audit trail untuk tracking aktivitas admin	Semua tindakan admin tercatat untuk transparansi
18	KNF18	Email notification menggunakan SMTP	Pengiriman email reset password dan notifikasi status
19	KNF19	Soft delete untuk data histori	Data dapat dihapus namun tetap tersimpan untuk backup
20	KNF20	Sistem dapat diakses 24/7	Ketersediaan sistem untuk pendaftaran kapan saja

3.5. Perancangan Sistem Database

Perancangan sistem database merupakan tahap penting dalam pengembangan backend sistem pendaftaran magang dan penelitian berbasis web. Database berfungsi sebagai media penyimpanan seluruh data sistem, seperti data pengguna, pengajuan, dokumen persyaratan, notifikasi, dan histori aktivitas. Perancangan database yang baik diperlukan untuk menjaga integritas data, meningkatkan efisiensi akses, serta memudahkan pengembangan dan pemeliharaan sistem di masa mendatang.

Pada proyek ini, perancangan database menggunakan pendekatan relasional dengan MySQL sebagai Database Management System (DBMS). Pendekatan ini dipilih karena mampu mengelola hubungan antar data secara terstruktur, mendukung transaksi ACID, serta memiliki kompatibilitas yang baik dengan framework Laravel sebagai backend sistem.
Proses perancangan database diawali dengan analisis kebutuhan data berdasarkan hasil survei dan diskusi dengan stakeholder di Dinas Kesehatan Kota Surabaya. Dari analisis tersebut, ditentukan entitas utama dalam sistem, yaitu User, Profil, Penelitian, Magang, Pengumuman, Notifikasi, dan Histori.

Setelah itu, ditentukan atribut dan tipe data pada setiap entitas sesuai dengan kebutuhan sistem. Sebagai contoh, tabel User memiliki atribut id_user sebagai primary key, username, email, password terenkripsi, serta role untuk membedakan admin dan mahasiswa. Tahap selanjutnya adalah penentuan relasi antar tabel, seperti one-to-one antara User dan Profil, serta one-to-many antara User dengan Penelitian dan Magang, serta antara pengajuan dengan Notifikasi dan Histori. Relasi ini bertujuan untuk menjaga konsistensi dan efisiensi pengelolaan data.

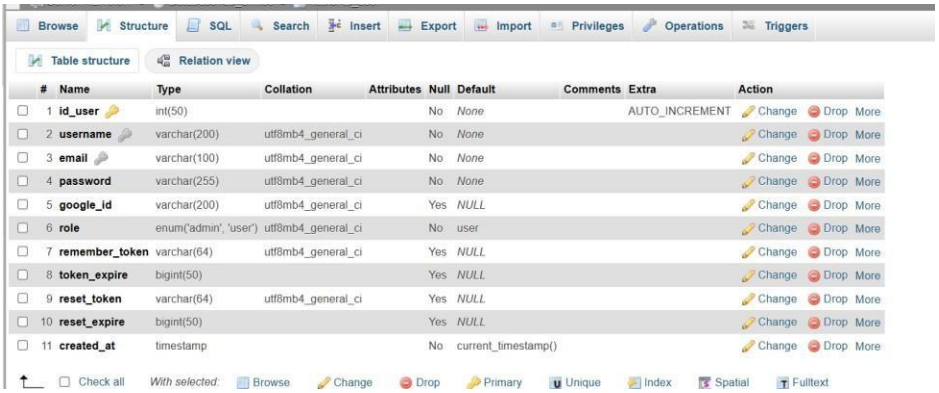
3.5.1. Implementasi Database

Gambar 3. 8 Struktur Database Pendaftaran Magang dan Penelitian

Pada gambar 3.8 merupakan tampilan struktur database sistem pendaftaran magang dan penelitian yang dikelola melalui phpMyAdmin. Database diberi nama db_dinkes dan terdiri dari 18 tabel yang terorganisir di sisi kiri panel navigasi. Setiap tabel dirancang dengan fungsi spesifik untuk mendukung operasional sistem backend secara menyeluruh.

Di bagian kiri gambar 3.8 terlihat daftar lengkap tabel-tabel yang terdapat dalam database, antara lain: cache, cache_locks, failed_jobs, jobs, job_batches, migrations, password_reset_tokens, sessions (tabel-tabel bawaan Laravel framework), serta tabel-tabel utama sistem yaitu tb_historiadmin, tb_historimhs, tb_magang, tb_notifadmin, tb_notifmhs, tb_penelitian, tb_pengumuman, tb_profil, tb_user, dan users dari framework Laravel.

3.5.1.1. Tabel User dan Admin



Gambar 3. 9 Struktur table user dan admin

Pada gambar 3.9 merupakan tampilan struktur tabel tb_user yang merupakan tabel paling krusial dalam sistem karena berfungsi sebagai pusat autentikasi dan otorisasi pengguna. Tabel ini menyimpan seluruh informasi akun pengguna baik admin maupun mahasiswa yang terdaftar dalam sistem pendaftaran magang dan penelitian.

3.5.1.2. Tabel Pendaftaran Magang

Gambar 3. 10 Struktur tabel pendaftaran magang

Pada gambar 3.10 merupakan tampilan struktur tabel tb_magang yang merupakan tabel inti dalam sistem untuk menyimpan seluruh data pengajuan magang atau kerja praktek yang diajukan oleh mahasiswa kepada Dinas Kesehatan Kota Surabaya. Tabel ini terdiri dari 15 kolom yang dirancang untuk menampung informasi lengkap mulai dari identitas pemohon, detail kegiatan magang, dokumen persyaratan, hingga status verifikasi dari admin.

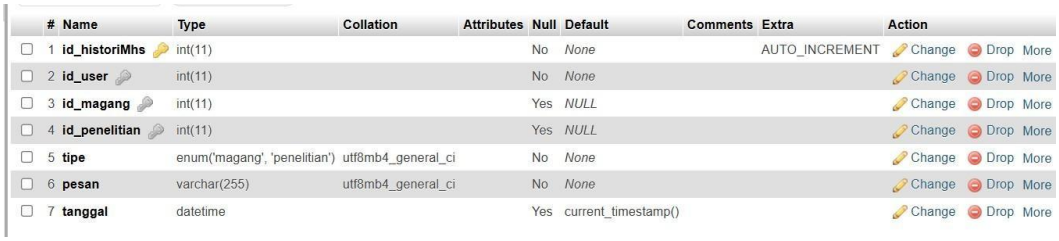
3.5.1.3. Tabel Pendaftaran Penelitian

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_penelitian	int(50)			No	None		AUTO_INCREMENT	Change Drop More
2	id_user	int(50)			No	None			Change Drop More
3	nama	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
4	nim	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
5	sekolah	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
6	jurusan	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
7	alamat	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
8	tujuan	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
9	tema	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
10	mulai	date			No	None			Change Drop More
11	berakhir	date			No	None			Change Drop More
12	upload_file	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change Drop More
13	tanggal_pengajuan	timestamp			No	current_timestamp()			Change Drop More
14	status	enum('pending', 'diterima', 'ditolak')	utf8mb4_general_ci		Yes	pending			Change Drop More
15	keterangan	text	utf8mb4_general_ci		Yes	NULL			Change Drop More
16	alasan	text	utf8mb4_general_ci		Yes	NULL			Change Drop More

Gambar 3. 11 Struktur tabel pendaftaran penelitian

Pada gambar 3.11 merupakan tampilan struktur tabel tb_penelitian yang merupakan tabel inti dalam sistem untuk menyimpan seluruh data pengajuan penelitian yang diajukan oleh mahasiswa kepada Dinas Kesehatan Kota Surabaya. Tabel ini terdiri dari 16 kolom yang dirancang untuk menampung informasi lengkap mulai dari identitas peneliti, detail penelitian, periode pelaksanaan, dokumen proposal, hingga status verifikasi dari admin.

3.5.1.4. Tabel Histori Pengguna

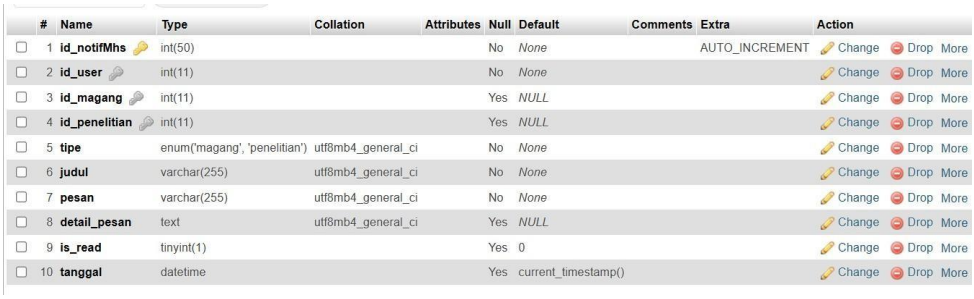


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_historiMhs	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	id_user	int(11)			No	None			Change Drop More
3	id_magang	int(11)			Yes	NULL			Change Drop More
4	id_penelitian	int(11)			Yes	NULL			Change Drop More
5	tipe	enum('magang', 'penelitian')	utf8mb4_general_ci		No	None			Change Drop More
6	pesan	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
7	tanggal	datetime			Yes	current_timestamp()			Change Drop More

Gambar 3. 12 Struktur Tabel Histori untuk Pengguna

Pada gambar 3.12 merupakan tampilan struktur tabel tb_historimhs yang berfungsi untuk menyimpan riwayat seluruh aktivitas yang dilakukan oleh mahasiswa dalam sistem pendaftaran magang dan penelitian. Tabel ini dirancang sebagai sistem logging yang mencatat setiap aksi mahasiswa, mulai dari pengajuan baru, perubahan data, hingga pembatalan pengajuan, sehingga memberikan transparansi dan dokumentasi lengkap untuk keperluan audit dan tracking aktivitas pengguna.

3.5.1.5. Tabel Notifikasi Pengguna



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_notifMhs	int(50)			No	None		AUTO_INCREMENT	Change Drop More
2	id_user	int(11)			No	None			Change Drop More
3	id_magang	int(11)			Yes	NULL			Change Drop More
4	id_penelitian	int(11)			Yes	NULL			Change Drop More
5	tipe	enum('magang', 'penelitian')	utf8mb4_general_ci		No	None			Change Drop More
6	judul	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
7	pesan	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
8	detail_pesan	text	utf8mb4_general_ci		Yes	NULL			Change Drop More
9	is_read	tinyint(1)			Yes	0			Change Drop More
10	tanggal	datetime			Yes	current_timestamp()			Change Drop More

Gambar 3. 13 Struktur Tabel Notifikasi Pengguna

Pada gambar 3.13 merupakan tampilan struktur tabel tb_notifmhs yang berfungsi untuk menyimpan seluruh notifikasi yang diterima oleh mahasiswa terkait status pengajuan magang dan penelitian mereka. Tabel ini dirancang sebagai sistem notifikasi real-time yang memberikan informasi kepada mahasiswa setiap kali terjadi perubahan status pengajuan, baik itu persetujuan, penolakan, maupun permintaan revisi dari admin Dinas Kesehatan Kota Surabaya.

3.5.1.6. Tabel Profil Pengguna

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_profil	int(50)			No	None		AUTO_INCREMENT	Change Drop More
2	id_user	int(50)			No	None			Change Drop More
3	nama_lengkap	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
4	nim	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
5	asal_sekolah	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
6	jurusan	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
7	email	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
8	no_telepon	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More

Gambar 3. 14 Struktur Tabel Profil pada pengguna

Pada gambar 3.14 di atas merupakan tampilan struktur tabel tb_profil yang berfungsi untuk menyimpan informasi lengkap profil mahasiswa yang terdaftar dalam sistem pendaftaran magang dan penelitian. Tabel ini dirancang terpisah dari tabel tb_user untuk menerapkan prinsip normalisasi database, dimana data autentikasi (username, password, role) dipisahkan dari data profil detail pengguna, sehingga memudahkan pengelolaan dan meningkatkan keamanan sistem.

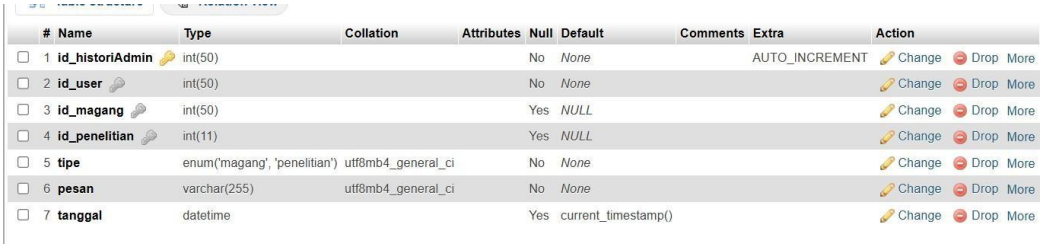
3.5.1.7. Tabel Pengumuman

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_pengum	int(50)			No	None			Change Drop More
2	judul	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	isi	text	utf8mb4_general_ci		No	None			Change Drop More
4	tanggal	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

Gambar 3. 15 Struktur Tabel Pengumuman Admin untuk Pengguna

Pada gambar 3.15 merupakan tampilan struktur tabel tb_pengumuman yang berfungsi untuk menyimpan seluruh pengumuman yang dibuat dan dipublikasikan oleh admin Dinas Kesehatan Kota Surabaya kepada mahasiswa pengguna sistem. Tabel ini dirancang sebagai sarana komunikasi satu arah dari admin kepada seluruh pengguna untuk menyampaikan informasi penting terkait program magang dan penelitian, perubahan kebijakan, jadwal penerimaan, atau informasi lainnya yang perlu diketahui oleh mahasiswa.

3.5.1.8. Tabel Histori Admin

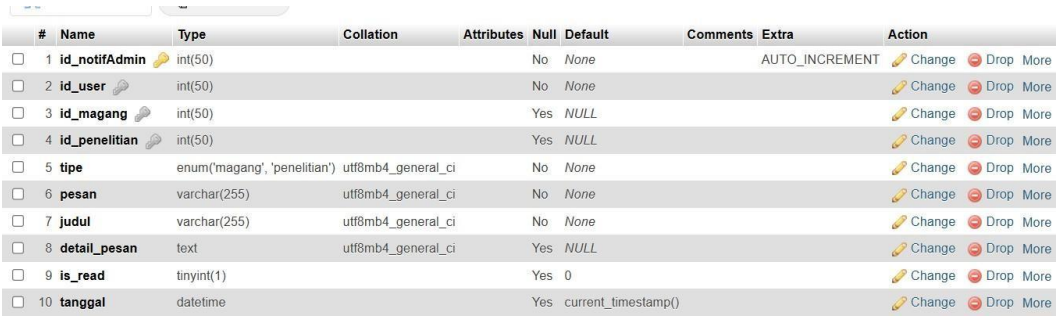


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_historiAdmin	int(50)			No	None		AUTO_INCREMENT	Change Drop More
2	id_user	int(50)			No	None			Change Drop More
3	id_magang	int(50)			Yes	NULL			Change Drop More
4	id_penelitian	int(11)			Yes	NULL			Change Drop More
5	tipe	enum('magang', 'penelitian')	utf8mb4_general_ci		No	None			Change Drop More
6	pesan	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
7	tanggal	datetime			Yes	current_timestamp()			Change Drop More

Gambar 3. 16 Struktur Tabel Histori Pada admin

Pada gambar 3.16 merupakan tampilan struktur tabel tb_historiadmin yang berfungsi untuk menyimpan riwayat seluruh aktivitas yang dilakukan oleh admin dalam mengelola dan memverifikasi pengajuan magang serta penelitian mahasiswa. Tabel ini dirancang sebagai sistem audit trail yang mencatat setiap tindakan admin, seperti menyetujui pengajuan, menolak pengajuan, atau memberikan keterangan, sehingga memberikan transparansi dan akuntabilitas dalam proses verifikasi yang dilakukan oleh Dinas Kesehatan Kota Surabaya.

3.5.1.9. Tabel Notifikasi Admin



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_notifAdmin	int(50)			No	None		AUTO_INCREMENT	Change Drop More
2	id_user	int(50)			No	None			Change Drop More
3	id_magang	int(50)			Yes	NULL			Change Drop More
4	id_penelitian	int(50)			Yes	NULL			Change Drop More
5	tipe	enum('magang', 'penelitian')	utf8mb4_general_ci		No	None			Change Drop More
6	pesan	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
7	judul	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
8	detail_pesan	text	utf8mb4_general_ci		Yes	NULL			Change Drop More
9	is_read	tinyint(1)			Yes	0			Change Drop More
10	tanggal	datetime			Yes	current_timestamp()			Change Drop More

Gambar 3. 17 Struktur Tabel Notifikasi Admin

Pada gambar 3.17 di atas merupakan tampilan struktur tabel tb_notifadmin yang berfungsi untuk menyimpan seluruh notifikasi yang diterima oleh admin terkait aktivitas mahasiswa dalam sistem pendaftaran magang dan penelitian. Tabel ini dirancang sebagai sistem notifikasi real-time yang memberikan informasi kepada admin setiap kali ada pengajuan baru dari mahasiswa, perubahan data pengajuan, atau pembatalan pengajuan, sehingga admin dapat segera melakukan verifikasi dan memberikan respon yang cepat kepada mahasiswa.

3.6. Hasil Dan Implementasi

Adapun gambaran terkait implementasi sistem backend yang dilakukan selama periode kerja praktek pada Dinas Kesehatan Kota Surabaya, meliputi proses perancangan, pengembangan, dan pengelolaan sistem pendaftaran magang dan penelitian berbasis web. Sistem ini dibangun menggunakan framework Laravel sebagai backend utama yang berfungsi mengelola logika bisnis, validasi data, manajemen database, serta integrasi dengan antarmuka pengguna.

Backend sistem dirancang untuk mempermudah pengelolaan data pengajuan mahasiswa, verifikasi dokumen oleh admin, sistem notifikasi otomatis, serta pengelolaan pengumuman yang terintegrasi dalam satu platform terpadu.

Implementasi backend mencakup pengembangan struktur database relasional yang terorganisir, pembuatan controller dan routing untuk menangani berbagai request HTTP, implementasi sistem autentikasi dan otorisasi berbasis role (admin dan mahasiswa), serta integrasi file upload untuk dokumen persyaratan pengajuan. Sistem backend juga dilengkapi dengan fitur notifikasi real-time yang memberikan feedback kepada mahasiswa setiap kali status pengajuan diubah oleh admin, serta sistem histori yang mencatat seluruh aktivitas pengguna untuk keperluan audit dan transparansi.

3.6.1. Halaman Route Web

Sistem routing dalam aplikasi pendaftaran magang dan penelitian ini menggunakan Laravel Framework versi 12 sebagai fondasi utama pengembangan backend. Laravel dipilih karena menyediakan sistem routing yang powerful, fleksibel, dan mudah dipahami dengan sintaks yang ekspresif. Routing dalam Laravel berfungsi sebagai peta navigasi yang menghubungkan setiap HTTP request dari browser pengguna dengan method controller yang tepat untuk memproses request tersebut. File routing utama aplikasi ini adalah `routes/web.php` yang menjadi single source of truth untuk seluruh route yang tersedia dalam sistem.

Struktur routing dirancang dengan pendekatan role-based architecture yang membedakan akses dan fungsi berdasarkan peran pengguna dalam sistem. Terdapat tiga kategori utama pengguna: Guest (pengunjung belum login), User/Mahasiswa (mahasiswa terdaftar), dan Admin (pengelola sistem dari Dinas Kesehatan). Setiap kategori memiliki set route berbeda dengan tingkat akses sesuai kebutuhan dan tanggung jawab mereka. Pendekatan ini meningkatkan keamanan sistem dan membuat kode lebih terorganisir serta mudah di-maintain.

Sistem routing menerapkan secara konsisten di seluruh aplikasi. Named routes memungkinkan setiap route diberi nama unik sebagai referensi di view blade, controller, atau middleware. Keuntungannya adalah jika URL perlu diubah, developer hanya mengubah definisi route di `web.php`, sementara semua referensi tetap bekerja tanpa perubahan. Konvensi penamaan mengikuti pattern konsisten: route admin diberi prefix admin diikuti nama fungsi.

.

Routing mendukung route parameters untuk membuat endpoint dinamis dan reusable. menggunakan parameter `{id_magang}` yang diisi dengan ID pengajuan berbeda. Laravel otomatis mengekstrak nilai dari URL dan mengirimkannya ke method controller untuk query database. Pendekatan ini lebih efisien dibanding membuat route terpisah untuk setiap ID dan membuat URL lebih semantic

Untuk meningkatkan performa, routing memanfaatkan route caching Laravel yang mengkompilasi semua definisi route menjadi satu file cache. Di production server, command `php artisan route:cache` dapat dijalankan untuk generate cache file yang di-load sangat cepat, mengurangi overhead parsing `web.php` setiap request. Ini bermanfaat untuk aplikasi dengan banyak route seperti sistem ini yang memiliki lebih dari 40 endpoint.

Sistem juga mempertimbangkan backward compatibility dengan beberapa route redundant. Misalnya Beranda dan user beranda keduanya mengarah ke *BerandaController* untuk memastikan link lama tetap berfungsi meskipun struktur URL berubah. Ini untuk menghindari broken links dan menjaga user experience selama proses refactoring.

3.6.1.1. Route Import Namespace dan Class Controller

```php
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Middleware\profil_guard;

use App\Http\Controllers\userController\RegistrasiController;
use App\Http\Controllers\userController\LoginController;
use App\Http\Controllers\userController\BerandaController;
use App\Http\Controllers\userController\UserProfilController;
use App\Http\Controllers\userController\HistoriUserController;
use App\Http\Controllers\userController\NotifUserController;
use App\Http\Controllers\userController\PengumumanUserController;
use App\Http\Controllers\userController\FormMagangController;
use App\Http\Controllers\userController\FormPenelitianController;
use App\Http\Controllers\userController\PasswordResetController;


use App\Http\Controllers\adminController\BerandaAdminController;
use App\Http\Controllers\adminController\PengumumanAdminController;
use App\Http\Controllers\adminController\HistoriAdminController;
use App\Http\Controllers\adminController\NotifAdminController;
use App\Http\Controllers\adminController\DaftarMagangController;
use App\Http\Controllers\adminController\DetailMagangController;
use App\Http\Controllers\adminController\MagangController;
use App\Http\Controllers\adminController\DaftarPenelitianController;
use App\Http\Controllers\adminController\DetailPenelitianController;
use App\Http\Controllers\adminController\PenelitianController;
```

Gambar 3. 18 Import dari setiap Controller

Pada gambar 3.18 merupakan tampilan section import dependencies di bagian paling atas file routes/web.php yang berfungsi sebagai deklarasi seluruh controller dan middleware yang akan digunakan dalam sistem routing. Di baris pertama terdapat import Illuminate\Support\Facades\Route yang merupakan facade Laravel untuk mendefinisikan routing, memberikan akses ke method-method routing seperti get(), post(), delete(), dan group(). Tepat di bawahnya terdapat import middleware custom App\Http\Middleware\profil_guard yang dirancang untuk proteksi akses halaman profil, meskipun dalam implementasi actual middleware ini tidak digunakan karena sistem menggunakan session-based authentication manual.

Selanjutnya terdapat blok import controller untuk fitur mahasiswa yang dikelompokkan dalam namespace App\Http\Controllers\userController. Import ini mencakup sembilan controller utama: RegistrasiController untuk menangani pendaftaran akun mahasiswa baru dengan validasi data dan penyimpanan ke database, LoginController untuk proses autentikasi pengguna dengan pencocokan email dan password, BerandaController untuk menampilkan dashboard mahasiswa dengan statistik pengajuan, UserProfilController untuk kelola data profil lengkap mahasiswa, HistoriUserController untuk menampilkan riwayat semua pengajuan yang pernah dibuat, NotifUserController untuk sistem notifikasi real-time mahasiswa, PengumumanUserController untuk menampilkan pengumuman dari admin, FormMagangController untuk memproses pengajuan magang termasuk validasi dan upload file, FormPenelitianController untuk memproses pengajuan penelitian, dan PasswordResetController untuk fitur reset password via email. Di bawah import controller mahasiswa terdapat blok import controller untuk fitur admin yang dikelompokkan dalam namespace App\Http\Controllers\adminController. Import ini mencakup sepuluh controller administratif: BerandaAdminController untuk dashboard admin dengan statistik lengkap sistem, PengumumanAdminController untuk CRUD pengumuman yang akan dilihat mahasiswa, HistoriAdminController untuk riwayat aktivitas admin seperti verifikasi pengajuan, NotifAdminController untuk notifikasi admin ketika ada pengajuan baru, DaftarMagangController untuk menampilkan tabel daftar semua pengajuan magang dengan pagination dan filter, DetailMagangController untuk menampilkan detail lengkap satu pengajuan magang, MagangController untuk operasi CRUD dan update status pengajuan magang, DaftarPenelitianController untuk daftar pengajuan penelitian, DetailPenelitianController untuk detail pengajuan penelitian, dan PenelitianController untuk operasi CRUD penelitian.

Pemisahan import controller berdasarkan namespace ini mengikuti prinsip separation of concerns dan single responsibility principle, dimana setiap controller memiliki tanggung jawab spesifik dan mudah ditemukan berdasarkan kategori fungsinya. Struktur ini memudahkan developer untuk navigasi kode ketika perlu melakukan maintenance atau menambah fitur baru, karena cukup melihat namespace untuk mengetahui controller mana yang menangani fitur mahasiswa atau admin.

3.6.1.2. User Web Route

```php
//user
// UserProfil routes (tanpa middleware auth karena kita pakai session manual)
Route::get('/userProfil', [UserProfilController::class, 'index'])->name('userProfil');
Route::post('/userProfil/update', [UserProfilController::class, 'store'])->name('userProfil.update');

// User routes
Route::prefix('user')->name('user.')->group(function () {
    Route::get('/beranda', [BerandaController::class, 'index'])->name('beranda');
    Route::get('/userProfil', [UserProfilController::class, 'index'])->name('profil');
    Route::post('/userProfil/update', [UserProfilController::class, 'store'])->name('profil.update');
});

// Beranda routes (untuk kompatibilitas dengan blade template)
Route::get('/beranda', [BerandaController::class, 'index'])->name('beranda');

//Pengumuman user
Route::get('/pengumumanUser', [PengumumanUserController::class, 'index'])->name('pengumumanUser');

//Form magang user
Route::get('/formMagang', [FormMagangController::class, 'index'])->name('formMagang');
Route::post('/formMagang', [FormMagangController::class, 'store'])->name('formMagang.store');

//Form penelitian user
Route::get('/formPenelitian', [FormPenelitianController::class, 'index'])->name('formPenelitian');
Route::post('/formPenelitian', [FormPenelitianController::class, 'store'])->name('formPenelitian.store');

//Histori user
Route::get('/historiUser', [HistoriUserController::class, 'index'])->name('historiUser');

//Notif user
Route::get('/notifications', [NotifUserController::class, 'index'])->name('notifUser');
Route::get('/notifications/unread-count', [NotifUserController::class, 'getUnreadCount'])->name('notifications.unreadCount');
Route::get('/notifications/check-new', [NotifUserController::class, 'checkNewNotifications'])->name('notifications.checkNew');
Route::post('/notifications/mark-read/{id}', [NotifUserController::class, 'markRead'])->name('notifications.markRead');
```

Gambar 3. 19 Controller untuk User

Section route autentikasi menjadi pintu masuk utama mahasiswa ke dalam sistem. Di bagian paling atas terdapat default route dengan method get pada root URL yang secara otomatis mengarahkan pengunjung ke halaman login menggunakan fungsi redirect ke route login ketika mengakses alamat utama website, memastikan setiap pengunjung yang membuka website akan langsung diarahkan ke halaman login sebagai starting point sistem. Tepat di bawahnya terdapat dua route registrasi yaitu route pertama dengan method get pada URL registrasi yang terhubung ke method showForm di RegistrasiController untuk menampilkan form pendaftaran dengan field nama lengkap, email, password, dan konfirmasi password, sedangkan route kedua dengan method post pada URL registrasi bernama registrasi.store terhubung ke method store yang memproses data registrasi, melakukan validasi input menggunakan Laravel validation rules dimana email harus valid dan unique di database, password minimal 8 karakter, password confirmation harus match, mengenkripsi password menggunakan bcrypt, menyimpan data ke tabel tb_user dengan role default user, dan redirect ke halaman login dengan success message.

Di bawah route registrasi terdapat tiga route login yaitu route pertama dengan method get pada URL login menampilkan halaman login dengan form input email dan password, route kedua dengan method post memproses autentikasi dengan validasi kredensial, pembuatan session yang menyimpan email, id_user, dan role, serta redirect ke dashboard sesuai role pengguna, sedangkan route ketiga dengan method get pada URL logout menghancurkan session dan mengarahkan kembali ke halaman login. Section autentikasi ditutup dengan empat route password reset yaitu route pertama menampilkan form input email untuk request reset password, route kedua memvalidasi email dan mengirim link reset via email, route ketiga menampilkan form input password baru dengan validasi token, dan route keempat memproses perubahan password dengan update database dan redirect ke login dengan diterima.

Route untuk dashboard dan profil mahasiswa dimulai dengan route method get pada URL beranda yang terhubung ke BerandaController method index yang mengambil data pengajuan milik mahasiswa dari database berdasarkan session id_user, menghitung jumlah pengajuan berdasarkan status yaitu pending, diterima, dan ditolak menggunakan query builder Laravel, mengambil 5 notifikasi terbaru yang belum dibaca dari tabel tb_notifmhs, dan mengirim data ke view beranda dalam bentuk compact variable atau array. Di bawahnya terdapat route group dengan prefix user yang membungkus beberapa route mahasiswa untuk konsistensi URL, di mana route dengan URL user beranda juga mengarah ke BerandaController untuk kompatibilitas dengan struktur URL yang berbeda, dan route dengan URL user userProfil menampilkan halaman profil mahasiswa melalui method index di UserProfilController yang melakukan join query antara tabel tb_user dan tb_profil berdasarkan id_user, mengambil data lengkap profil seperti nama, NIM, asal sekolah, jurusan, nomor telepon, dan mengirimnya ke view profil.

Route untuk form pengajuan magang dan penelitian dimulai dengan route method get pada URL formMagang yang menampilkan form pengajuan dengan field nama pemohon, NIM, asal sekolah, jurusan, email, tanggal mulai dan berakhir magang, serta upload field untuk proposal dan surat pengantar, method ini juga mengambil data profil mahasiswa untuk auto-fill beberapa field form. Route dengan method post pada URL formMagang memproses submit pengajuan melalui method store yang melakukan validasi input form dengan Laravel validation rules, setelah validasi berhasil melakukan upload file ke folder storage dengan generate nama file unik, menyimpan path file dan data pengajuan ke tabel tb_magang dengan status default pending, membuat notifikasi baru untuk admin di tabel tb_notifadmin, menyimpan histori aktivitas di tabel tb_historimhs, dan mengembalikan redirect ke halaman histori dengan flash message success.

Route untuk fitur pengumuman, histori, dan notifikasi mahasiswa dimulai dengan route method get pada URL pengumumanUser yang mengambil semua data pengumuman dari tabel tb_pengumuman, mengurutkannya berdasarkan tanggal terbaru, dan menampilkannya dalam bentuk card yang berisi judul, isi singkat, dan tanggal publikasi. Route method get pada URL historiUser mengambil riwayat semua pengajuan mahasiswa dari tabel tb_magang dan tb_penelitian menggunakan union query untuk menggabungkan data, menambahkan kolom tipe untuk membedakan jenis pengajuan, dan menampilkan dalam tabel dengan kolom nomor, jenis pengajuan, tanggal pengajuan, status dengan badge warna sesuai kondisi, serta keterangan jika ada, dilengkapi fitur filter berdasarkan jenis dan status pengajuan.

3.6.1.3. Admin Web Route

```php
// Admin routes
Route::prefix('admin')->name('admin.')->group(function () {
    Route::get('/berandaAdmin', [BerandaAdminController::class, 'index'])->name('berandaAdmin');

   // Pengumuman
Route::get('/pengumumanAdmin', [PengumumanAdminController::class, 'index'])->name('pengumumanAdmin');
Route::post('/pengumuman/store', [PengumumanAdminController::class, 'store'])->name('pengumuman.store');
Route::post('/pengumuman/update/{id}', [PengumumanAdminController::class, 'update'])->name('pengumuman.update');
Route::delete('/pengumuman/delete/{id}', [PengumumanAdminController::class, 'destroy'])->name('pengumuman.delete');


// Histori Admin
Route::get('/historiAdmin', [HistoriAdminController::class, 'index'])->name('historiAdmin');
Route::delete('/historiAdmin/delete/{id}', [HistoriAdminController::class, 'delete'])->name('historiAdmin.delete');
Route::post('/historiAdmin/bulk-delete', [HistoriAdminController::class, 'bulkDelete'])->name('historiAdmin.bulkDelete');

//Notif Admin
Route::get('/notif', [NotifAdminController::class, 'index'])->name('notifAdmin');
    Route::post('/notif/{id}/read', [NotifAdminController::class, 'markRead'])->name('notif.read');
    Route::delete('/notif/{id}/delete', [NotifAdminController::class, 'delete'])->name('notif.delete');
    Route::post('/notif/bulk-delete', [NotifAdminController::class, 'bulkDelete'])->name('notif.bulkDelete');

//Daftar magang admin
Route::get('/daftar-magang', [DaftarMagangController::class, 'index'])->name('daftarMagang');
Route::get('/detail-magang/{id_magang}', [DetailMagangController::class, 'show'])->name('detailMagang');

Route::delete('/magang/{id_magang}', [MagangController::class, 'destroy'])->name('magang.destroy');
Route::post('/magang/{id_magang}/status', [MagangController::class, 'updateStatus'])->name('magang.updateStatus');
Route::get('/magang/{id_magang}/file', [MagangController::class, 'viewFile'])->name('magang.viewFile');
Route::delete('/magang/auto-delete', [MagangController::class, 'autoDelete'])->name('magang.autoDelete');


//Daftar penelitian admin
Route::get('/daftarPenelitian', [DaftarPenelitianController::class, 'index'])->name('daftarPenelitian');
Route::get('/detail-penelitian/{id_penelitian}', [DetailPenelitianController::class, 'show'])->name('detailPenelitian');

Route::delete('/penelitian/{id_penelitian}', [PenelitianController::class, 'destroy'])->name('penelitian.destroy');
Route::post('/penelitian/{id_penelitian}/status', [PenelitianController::class, 'updateStatus'])->name('penelitian.updateStatu
Route::get('/penelitian/{id_penelitian}/file', [PenelitianController::class, 'viewFile'])->name('penelitian.viewFile');
Route::delete('/penelitian/auto-delete', [PenelitianController::class, 'autoDelete'])->name('penelitian.autoDelete');
```

Gambar 3. 20 Controller untuk Admin

Pada gambar 3.20, Route group admin membungkus seluruh endpoint administratif dalam satu block dengan prefix admin dan name prefix admin, sehingga route yang didefinisikan sebagai berandaAdmin akan menjadi admin.berandaAdmin dan dapat diakses via URL admin berandaAdmin. Di dalam group, route pertama adalah dashboard admin dengan method get pada URL berandaAdmin yang terhubung ke BerandaAdminController method index,

Route untuk kelola pengumuman memberikan fungsi CRUD lengkap kepada admin dimana route dengan method get menampilkan daftar pengumuman dalam tabel dengan pagination dan action button, route dengan method post pada URL store memproses tambah pengumuman baru dengan validasi input dan menyimpan ke tabel tb_pengumuman, route dengan method post pada URL update memproses edit pengumuman berdasarkan ID, dan route dengan method delete menghapus pengumuman dari database, setiap aksi disertai dengan penyimpanan histori admin dan mengembalikan untuk ditampilkan menggunakan SweetAlert.

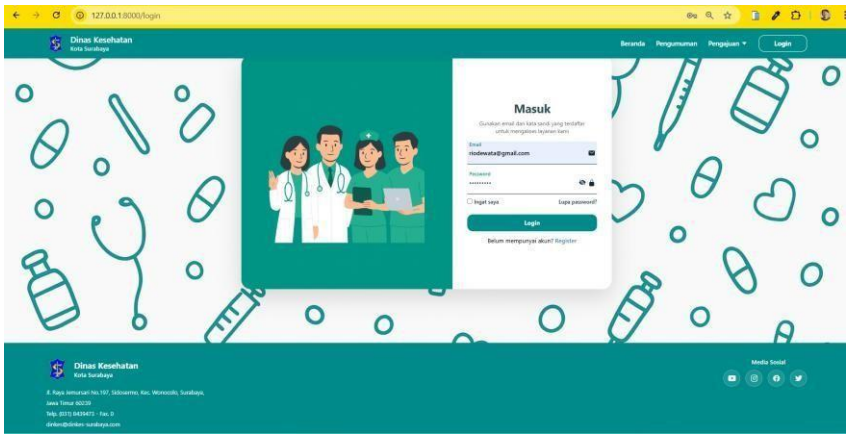
Route untuk kelola pengajuan magang dan penelitian memiliki struktur identik dimana route dengan method get menampilkan daftar pengajuan dalam tabel dengan pagination, filter status, dan search box, route detail dengan parameter ID menampilkan informasi lengkap pengajuan beserta data mahasiswa dan file dokumen yang dapat diunduh. Route paling krusial adalah route dengan method post untuk update status yang memproses verifikasi pengajuan dengan menerima AJAX request berisi status baru yaitu diterima atau ditolak serta keterangan dari admin, method ini melakukan validasi input, update field status dan keterangan di database, membuat notifikasi baru untuk mahasiswa di tabel tb_notifmhs, menyimpan histori aktivitas admin, dan memberikan response untuk ditampilkan SweetAlert. Route tambahan meliputi route untuk view file dokumen dengan validasi file exist dan return response dengan header yang sesuai, route untuk hapus pengajuan beserta file dokumen dari storage, dan route utility untuk auto delete pengajuan yang ditolak lebih dari 30 hari sebagai maintenance database periodic, pada auto delete ini bersifat opsional, Karena jarang digunakan.

Route histori admin menampilkan riwayat aktivitas admin dari tabel tb_historiadmin dengan pagination dan fitur delete untuk cleanup histori lama, dilengkapi dengan bulk delete untuk menghapus multiple record sekaligus. Route notifikasi admin menampilkan daftar notifikasi pengajuan baru dari mahasiswa dengan pagination dan badge unread count, dilengkapi dengan endpoint untuk menandai notifikasi sudah dibaca, menghapus satu notifikasi, dan bulk delete untuk cleanup inbox admin secara efisien.

3.6.2. Implementasi Menu Login

3.6.2.1. Halaman Login

LoginController menangani proses autentikasi pengguna ke dalam sistem dengan fitur yang lebih kompleks termasuk session management, remember me functionality, dan token-based authentication untuk auto-login. Berikut gambar 3.21 hasil tampilan halaman login .



Gambar 3. 21 Halaman Login

3.6.2.2. Implementasi Login Controller

```php
/**
 * Proses login
 */
public function login(Request $request)
{
    $request->validate([
        'email'    => 'required|email',
        'password' => 'required',
    ]);

    $email    = $request->input('email');
    $password = $request->input('password');
    $remember = $request->filled('remember');

    // Cek user berdasarkan email
    $user = User::where('email', $email)->first();

    if ($user && Hash::check($password, $user->password)) {
        // Regenerate session ID untuk keamanan
        $request->session()->regenerate();

        // Set session
        session()->put([
            'email'    => $user->email,
            'role'     => $user->role,
            'user_id'  => $user->id_user,
            'username' => $user->username,
        ]);

        // Update last login time (optional)
        $user->update([
            'last_login' => Carbon::now()
        ]);

        // Handle "Ingat Saya"
        if ($remember) {
            $rememberToken = bin2hex(random_bytes(32));
            $tokenExpire   = time() + (30 * 24 * 60 * 60); // 30 hari

            $user->update([
                'remember_token' => $rememberToken,
                'token_expire'   => $tokenExpire,
            ]);

            $cookieName = $user->role === 'admin'
                ? 'admin_remember_token'
                : 'user_remember_token';

            Cookie::queue($cookieName, $rememberToken, 43200); // menit (30 hari)
        } else {
            // Hapus remember token jika user tidak centang remember me
            $user->update([
                'remember_token' => null,
                'token_expire' => null,
            ]);
        }

        // Redirect sesuai role
        if ($user->role === 'admin') {
            return redirect()->route('admin.berandaAdmin');
        }

        if ($user->role === 'user') {
            $hasProfile = Profil::where('id_user', $user->id_user)
                ->whereNotNull('nama_lengkap')
                ->where('nama_lengkap', '!=', '')
                ->exists();

         if (!$hasProfile) {
            return redirect()->route('user.profil');
        }

         return redirect()->route('user.beranda');
        }

    }

    return back()->with('login_error', 'Email atau password salah.');
}

/**
 * Cek remember me token (dipanggil di middleware/constructor kalau mau auto login)
 */
public function checkRememberToken(Request $request)
{
    $now = time();

    // cek admin cookie
    if ($request->hasCookie('admin_
```

Gambar 3. 22 Hasil Kode pada Login

Pada gambar 3.22 merupakan tampilan untuk Login, dengan menunjukkan alur autentikasi yang terdiri dari empat tahap utama.

Tahap pertama adalah validasi input untuk memastikan email dalam format yang benar dan password tidak kosong. Variable $remember menggunakan method filled ‘remember’ yang mengembalikan true jika checkbox "Ingat Saya" dicentang oleh pengguna.

Tahap kedua adalah verifikasi kredensial dimana sistem melakukan query ke database untuk mencari user berdasarkan email menggunakan Eloquent Model User::where('email', $email)->first(). Password kemudian diverifikasi menggunakan Hash::check() yang membandingkan password plaintext dari input dengan password hash di database. Algoritma bcrypt yang digunakan memastikan password tetap aman karena menggunakan hashing one-way yang tidak dapat di-decrypt.

Tahap ketiga adalah pembuatan session jika autentikasi berhasil. Langkah pertama adalah regenerate session ID menggunakan $request->session()->regenerate() untuk mencegah session fixation attack. Kemudian sistem membuat session baru menggunakan session()->put() yang menyimpan data penting seperti email, role untuk authorization, user_id sebagai foreign key, dan username untuk display. Data ini akan digunakan di seluruh aplikasi untuk identifikasi dan otorisasi user.

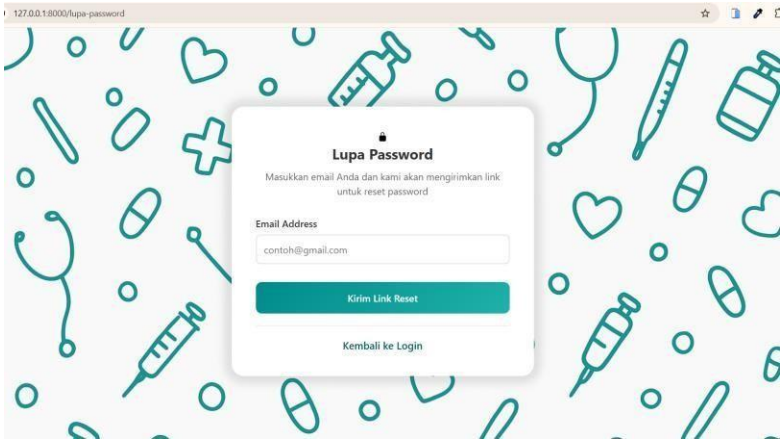
Tahap keempat adalah implementasi fitur "Ingat Saya" yang bersifat opsional. Jika user mencentang checkbox, sistem generate secure random token menggunakan bin2hex(random_bytes(32)) yang menghasilkan string hexadecimal 64 karakter. Token expire diset 30 hari dari sekarang dalam format unix timestamp. Token dan expire time disimpan ke database, kemudian cookie dibuat dengan nama berbeda untuk admin dan user menggunakan Cookie dengan durasi 43200 menit (30 hari).

Redirect dilakukan berdasarkan role dimana admin langsung diarahkan ke dashboard admin, sedangkan mahasiswa harus melewati pengecekan kelengkapan profil terlebih dahulu menggunakan method exists(). Jika profil belum lengkap, mahasiswa diarahkan ke halaman profil untuk mengisi data wajib. Ini memastikan setiap mahasiswa memiliki data profil lengkap sebelum dapat menggunakan sistem. Jika autentikasi gagal, sistem mengembalikan user ke halaman login dengan flash message error menggunakan back dengan login eror. Dengan Gambaran ini berikut gambar

3.6.3. Implementasi Reset Password

PasswordResetController menangani fitur reset password bagi user yang lupa kredensial mereka. Controller ini mengimplementasikan alur reset password yang aman melalui email verification dengan token-based authentication.

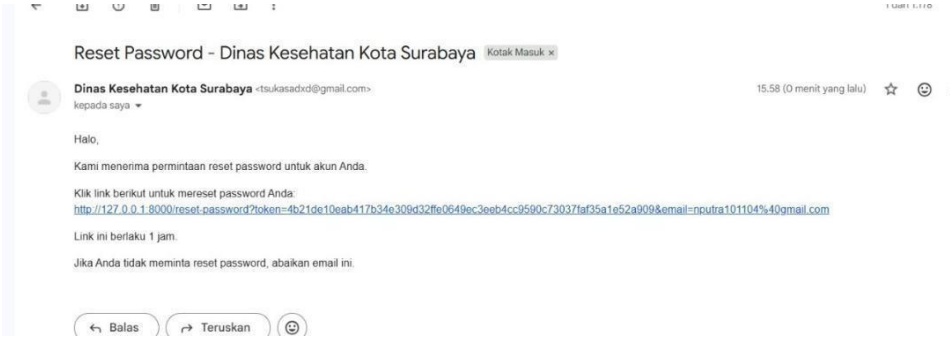
3.6.3.1. Halaman User Lupa Password



Gambar 3. 23 Tampilan Halaman Lupa Password

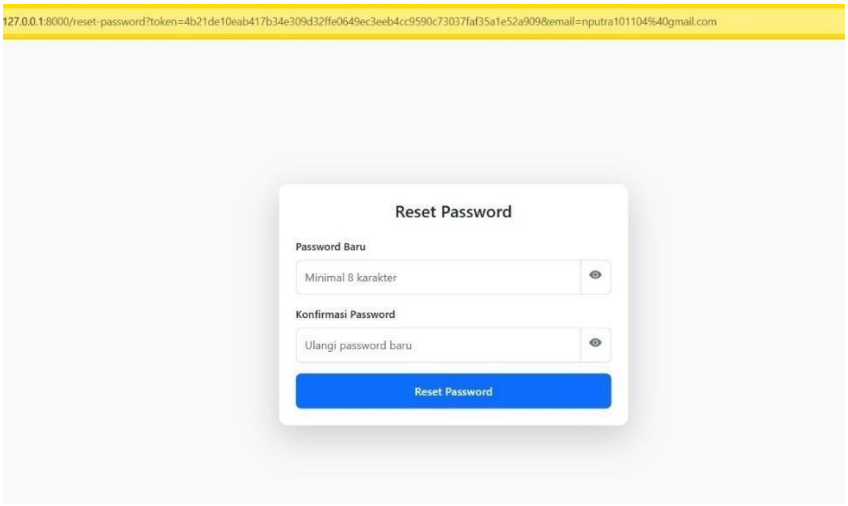
Pada gambar 3.23 menunjukkan tampilan halaman "Lupa Password" yang diakses ketika user mengklik link "Lupa password?" di halaman login. Halaman ini menampilkan form sederhana dengan background pattern ikon medis yang konsisten dengan tema Dinas Kesehatan. Form memiliki card putih di tengah dengan judul "Lupa Password" dan deskripsi "Masukkan email Anda dan kami akan mengirimkan link untuk reset password". Terdapat satu input field berlabel "Email Address" dengan placeholder contoh gmail yang membantu user memahami format email yang diharapkan. Tombol utama berwarna teal dengan teks "Kirim Link Reset" dan di bawahnya terdapat link "Kembali ke Login" untuk navigasi kembali jika user membatalkan proses reset

3.6.3.2. Halaman User Pada Gmail



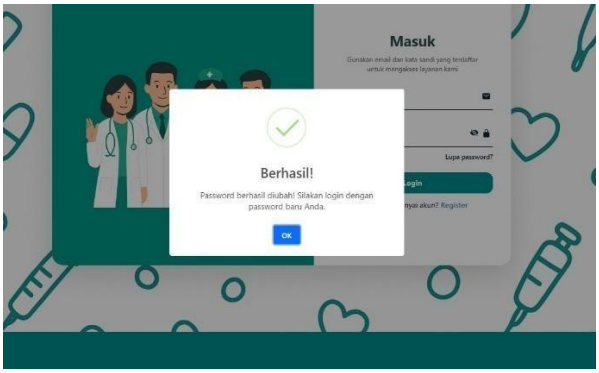
Gambar 3. 24 Url untuk menuju Halaman Update Password

Pada gambar 3.24 merupakan url untuk masuk dalam menu update password, seperti pada gambar 3.25.



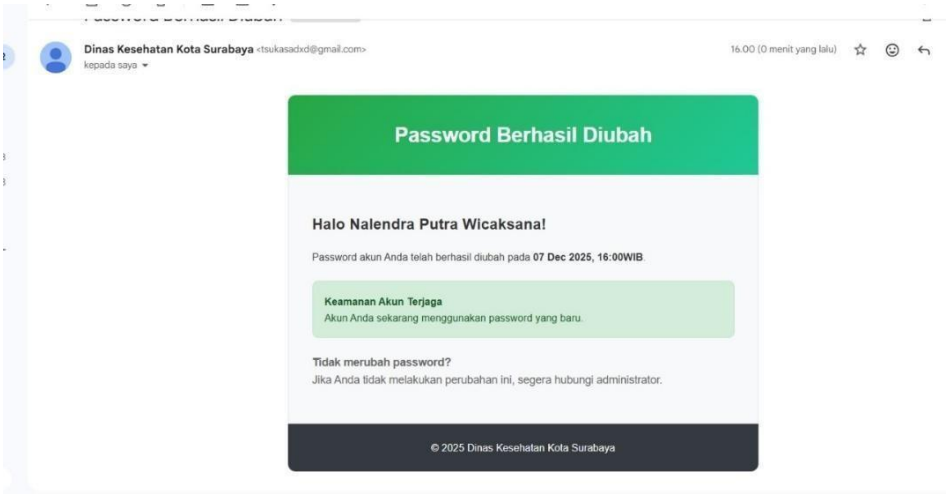
Gambar 3. 25 Halaman Reset Password

Pada gambar 3.25 menunjukkan halaman "Reset Password" yang muncul setelah user klik link dari email. URL bar menampilkan alamat lengkap dengan parameter token dan email yang panjang, menunjukkan bahwa link berhasil diproses. Halaman memiliki background putih bersih dengan card putih di tengah. Form memiliki judul "Reset Password" dan dua input field: "Password Baru" dengan placeholder "Minimal 8 karakter" dan "Konfirmasi Password" dengan placeholder "Ulangi password baru". Kedua field dilengkapi dengan icon mata di sebelah kanan untuk toggle show/hide password, meningkatkan usability karena user bisa memverifikasi password yang diketik. Jika sudah melewati ini maka akan muncul konfirmasi dengan sweetalert sesuai dengan gambar 3.26.



Gambar 3. 26 Password berhasil di rubah

Pada gambar 3.26 menunjukkan modal success "Berhasil!" dengan ikon centang hijau yang muncul di atas halaman login. Modal menampilkan pesan "Password berhasil diubah! Silakan login dengan password baru Anda." dengan tombol "OK" berwarna biru. Setalah itu maka akan muncul email untuk memverifikasi bahwa password telah di ubah sesuai pada gambar 3.26



Gambar 3. 27 Halaman Gmail Verifikasi pada Email

Pada gambar 3.27 menunjukkan email notifikasi yang diterima user setelah password berhasil direset. Email memiliki header dengan background hijau dan judul besar "Password Berhasil Diubah" yang immediately visible. Isi email menyapa user dengan nama "Halo Nalendra Putra Wicaksana!" yang personalized, membuat komunikasi lebih friendly. Email memberikan informasi spesifik "Password akun Anda telah berhasil diubah pada 07 Dec 2025, 16:00WIB" dengan timestamp yang exact, memungkinkan user untuk cross-check apakah perubahan dilakukan oleh mereka.

3.6.4. Implementasi PasswordResetController

```
 public function sendResetLink(Request $request)
    {
        $request->validate([
            'email' => 'required|email',
        ]);

        $email = $request->email;
        $user = User::where('email', $email)->first();

        if (!$user) {
            // Tetap tampilkan pesan sukses untuk keamanan
            return back()->with('success', 'Jika email terdaftar,
link reset telah dikirim.');
        }

        // Generate token
        $token = bin2hex(random_bytes(32));
        $expire = time() + 3600; // 1 jam

        // Update user
        $user->update([
            'reset_token' => $token,
            'reset_expire' => $expire,
        ]);

        // Buat reset link
        $resetLink = route('password.reset', [
            'token' => $token,
            'email' => $email
        ]);

        // Kirim email
        $emailSent = MailHelper::sendResetEmail($email, $resetLink);

        if ($emailSent) {
            return back()->with('success', 'Link reset password
telah dikirim ke email Anda.');
        } else {
            return back()->with('error', 'Gagal mengirim email.
Silakan coba lagi.');
        }
    }
```

Gambar 3. 28 Implementasi Pada reset paswword

Pada gambar 3.28 menunjukkan implementasi method sendResetLink() yang terdiri dari lima tahap pemrosesan yang secure dan terstruktur. Tahap pertama adalah validasi input menggunakan Laravel validation rules untuk memastikan field email wajib diisi dan dalam format email yang valid. Jika validasi gagal, Laravel otomatis mengembalikan user ke form dengan error message.

Tahap kedua adalah pencarian user di database menggunakan Query Builder Eloquent yang mencari record user berdasarkan email. Method first() mengembalikan object user jika ditemukan atau null jika tidak ada. Aspek keamanan penting terletak pada penanganan email yang tidak ditemukan. Jika user bernilai null (email tidak terdaftar), sistem tetap menampilkan pesan sukses yang identik dengan pesan untuk email yang terdaftar: "Jika email terdaftar, link reset telah dikirim." Ini adalah security best practice untuk mencegah email enumeration attack, dimana attacker bisa mencoba berbagai alamat email untuk mengetahui email mana yang terdaftar di sistem. Dengan pesan yang sama, attacker tidak bisa membedakan apakah email terdaftar atau tidak.

Tahap ketiga adalah generate reset token menggunakan fungsi bin2hex(random_bytes(32)). Fungsi random_bytes(32) menghasilkan 32 bytes data random yang cryptographically secure (256-bit entropy), kemudian bin2hex() mengkonversi bytes tersebut menjadi string hexadecimal sepanjang 64 karakter. Token ini sangat sulit ditebak karena memiliki kemungkinan kombinasi. Token expire diset menggunakan time 3600 dimana time() mengembalikan unix timestamp saat ini dan ditambah 3600 detik (1 jam). Durasi 1 jam dipilih sebagai balance antara keamanan (tidak terlalu lama agar token tidak bisa disalahgunakan) dan user experience (cukup waktu bagi user untuk cek email dan reset password).

Tahap keempat adalah menyimpan token ke database menggunakan Eloquent method update() yang mengupdate field reset_token dengan token yang baru di-generate dan reset_expire dengan timestamp kapan token akan expired. Data ini disimpan di tabel tb_user pada record user yang bersangkutan.

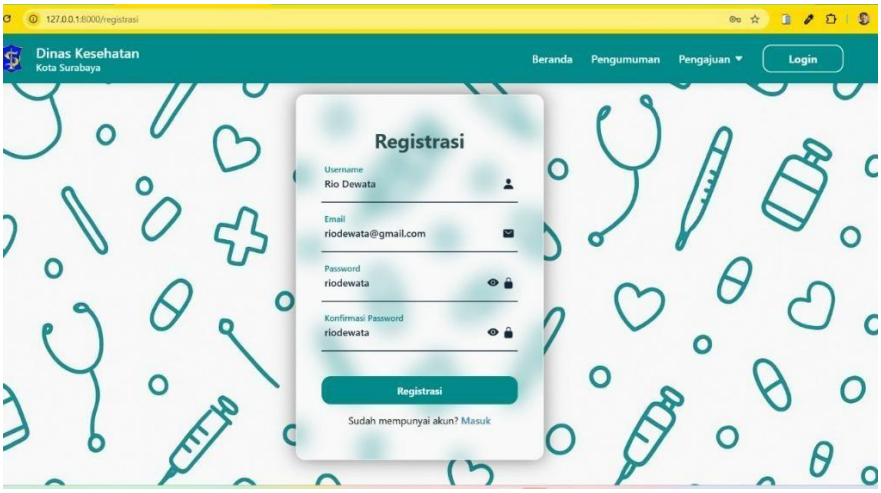
Tahap kelima adalah pembuatan reset link menggunakan Laravel helper route() yang generate URL lengkap ke route bernama 'password.reset' dengan menyertakan dua parameter: token dan email. Hasilnya adalah URL seperti http://domain.com/reset-password?token=abc123...&email=user@email.com. Parameter ini akan digunakan untuk validasi pada tahap berikutnya.

Tahap keenam adalah pengiriman email melalui custom helper class MailHelper. Helper ini menghandle konfigurasi SMTP, load template email HTML, replace placeholder dengan data actual (reset link, nama user), dan mengirim email. Method ini mengembalikan boolean true jika pengiriman berhasil atau false jika gagal karena alasan seperti SMTP error, invalid email, atau network issue.

Response handling dilakukan berdasarkan hasil pengiriman email. Jika berhasil, sistem menggunakan return back success untuk redirect kembali ke halaman form lupa password sambil membawa flash message sukses yang akan ditampilkan dalam modal atau alert. Jika gagal, sistem mengirim flash message error dengan instruksi untuk mencoba lagi, memberikan feedback yang clear kepada user bahwa proses belum selesai.

3.6.5. Halaman Register

Registrasi merupakan proses awal yang harus dilalui oleh calon pengguna sistem untuk membuat akun baru. RegistrasiController bertanggung jawab menangani proses pendaftaran akun mahasiswa baru ke dalam sistem dengan validasi yang ketat untuk memastikan integritas data.



Gambar 3. 29 Halaman registrasi

Gambar 3.29 menampilkan halaman registrasi pada URL /registrasi dengan desain khas Dinas Kesehatan, menggunakan background pattern ikon medis berwarna teal. Di tengah halaman terdapat card putih berisi form registrasi dengan judul “Registrasi” dan input Username, Email, Password, serta Konfirmasi Password yang dilengkapi fitur toggle visibility. Bagian bawah form terdapat tombol “Registrasi” berwarna teal dan teks “Sudah mempunyai akun? Masuk” yang menjadi link menuju halaman login, sehingga memudahkan navigasi bagi pengguna yang sudah memiliki akun.

3.6.6. Implementasi RegistrasiController

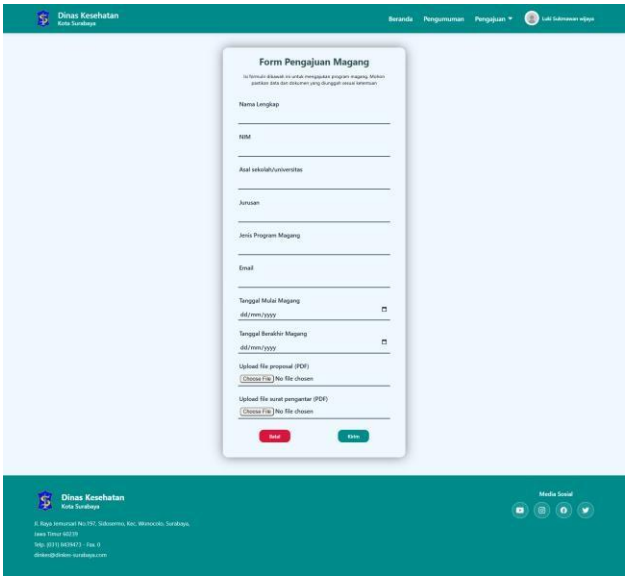
```
  public function showForm()
    {
        // Set unreadCount ke 0 untuk user yang belum login
        $unreadCount = 0;

        return view('user.registrasi', compact('unreadCount'));
    }

    public function store(Request $request)
    {
        $request->validate([
            'username' => 'required',
            'email'    => 'required|email',
            'password' => 'required|min:6',
        ]);

        // Cek email sudah ada atau belum
        $cekEmail = DB::table('tb_user')->where('email', $request-
>email)->first();

        if ($cekEmail) {
            return redirect()->back()->with('error_message', 'Email
sudah terdaftar, silakan gunakan email lain!');
        }

        // Insert ke database
        $insert = DB::table('tb_user')->insert([
            'username' => $request->username,
            'email'    => $request->email,
            'password' => Hash::make($request->password),
        ]);

        if ($insert) {
            // Simpan pesan sukses di session (belum redirect ke
login)
            return redirect()->back()->with('success_message',
'Registrasi berhasil, silakan login!');
        } else {
            // Simpan pesan error
            return redirect()->back()->with('error_message',
'Registrasi gagal, silakan coba lagi!');
        }
    }
}
```

Gambar 3. 30 Implementasi Register

Pada gambar 3.30 menunjukkan method store() yang memproses data registrasi dengan beberapa tahap validasi dan pemrosesan yang terstruktur. dimulai dari validasi input (username wajib, email harus valid, password minimal 6 karakter). Jika validasi gagal, user dikembalikan ke form beserta pesan error. Selanjutnya dilakukan pengecekan duplikasi email melalui query ke tabel tb_user; jika email sudah terdaftar, user diarahkan kembali dengan flash message error. Jika email valid, password di-hash menggunakan Hash lalu data disimpan ke database menggunakan insert(). Jika penyimpanan berhasil, sistem memberikan flash message sukses "Registrasi berhasil, silakan login!" dan mengarahkan user kembali ke halaman registrasi, sedangkan jika gagal ditampilkan pesan error. Sistem tidak langsung melakukan login otomatis demi keamanan dan memastikan user melakukan autentikasi eksplisit di halaman login.

3.6.7. Halaman Form Magang User

Form Pengajuan Magang merupakan halaman yang digunakan oleh mahasiswa untuk mengajukan permohonan magang ke Dinas Kesehatan Kota Surabaya. Halaman ini menampilkan sebuah formulir terpusat berisi data diri dan kebutuhan administrasi seperti nama lengkap, NIM, asal sekolah/universitas, jurusan, email, tanggal mulai dan berakhir magang, serta upload file proposal. Formulir ini dirancang agar mudah diisi dan membantu memastikan bahwa semua data dan dokumen yang dibutuhkan untuk proses pengajuan magang dapat dikumpulkan secara lengkap dan terstruktur.



Gambar 3. 31 Halaman Form Magang

Pada gambar 3.31 merupakan Halaman form pengajuan magang menampilkan tampilan yang bersih dan terstruktur dengan header bertema Dinas Kesehatan Kota Surabaya. Di bagian tengah terdapat form vertikal yang berisi input seperti Nama Lengkap, NIM, Asal Sekolah/Universitas, Jurusan, Email, tanggal mulai dan berakhir magang, serta upload file proposal. Pada form ini user akan mengirim pengajuan yang nantinya akan masuk ke admin.

3.6.8. Implementasi FormMagangController

```
            // Validasi form dengan pesan error khusus untuk PDF
 $request->validate([
            'nama'      => 'required|string|max:255',
            'nim'       => 'required|string|max:50',
            'sekolah'   => 'required|string|max:255',
            'jurusan'   => 'required|string|max:255',
            'jenis'     => 'required|string|max:255',
            'email'     => 'required|email|max:255',
            'mulai'     => 'required|date',
            'berakhir'  => 'required|date|after_or_equal:mulai',
            'proposal'  => 'required|file|mimes:pdf|max:10048',
            'pengantar' => 'required|file|mimes:pdf|max:10048',
        ], [
            // Custom error messages
            'proposal.required' => 'File proposal harus
diupload',
            'proposal.mimes' => 'File proposal harus berformat
```

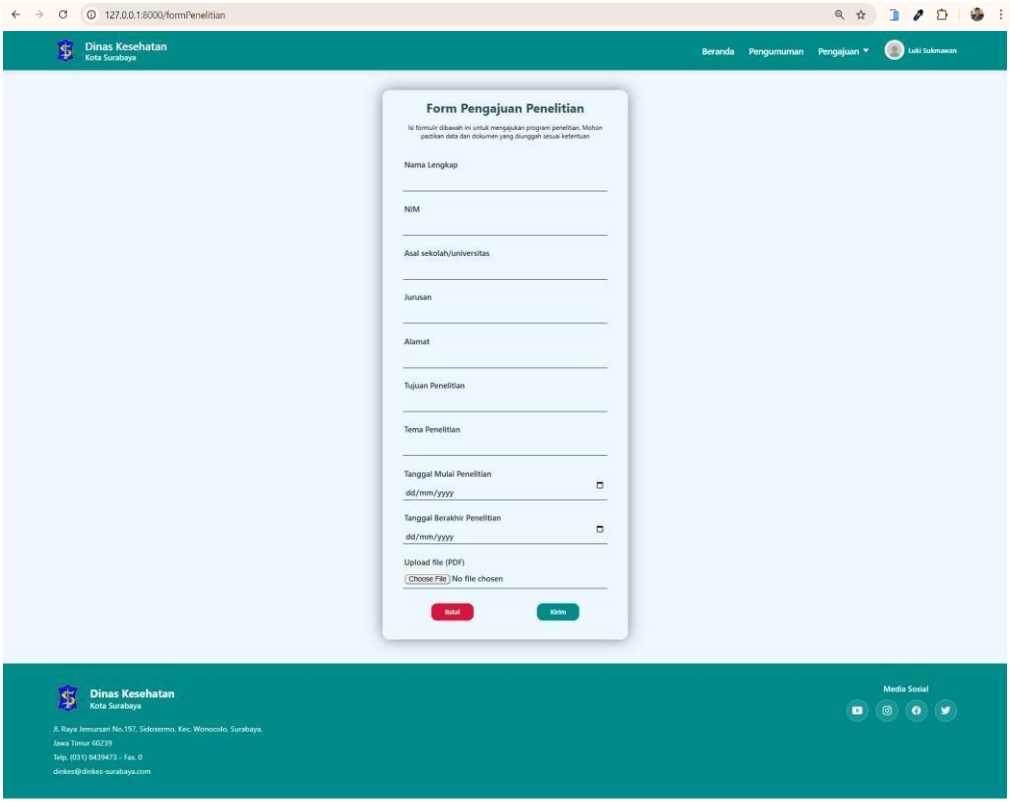
```
            'proposal.max' => 'Ukuran file proposal maksimal
10MB',
            'pengantar.required' => 'File surat pengantar harus
diupload',
            'pengantar.mimes' => 'File surat pengantar harus
berformat PDF',
            'pengantar.max' => 'Ukuran file surat pengantar
maksimal 10MB',
            'berakhir.after_or_equal' => 'Tanggal berakhir harus
sama atau setelah tanggal mulai',
        ]);

        // Simpan file
        $proposalPath = $request->file('proposal')->storeAs(
            'uploads',
            time().'_proposal.'.$request->file('proposal')-
>getClientOriginalExtension(),
            'public'
        );

        $pengantarPath = $request->file('pengantar')->storeAs(
            'uploads',
            time().'_pengantar.'.$request->file('pengantar')-
>getClientOriginalExtension(),
            'public'
        );

        // Simpan ke tabel tb_magang
           $id_magang = DB::table('tb_magang')->insertGetId([
            'id_user'    => $id_user,
            'nama'       => $request->nama,
            'nim'        => $request->nim,
            'sekolah'    => $request->sekolah,
            'jurusan'    => $request->jurusan,
            'jenis'      => $request->jenis,
            'email'      => $request->email,
            'mulai'      => $request->mulai,
            'berakhir'  => $request->berakhir,
            'proposal'  => basename($proposalPath),
            'pengantar' => basename($pengantarPath),
            'status'     => 'pending',
        ]);
```

Gambar 3. 32 Implementasi Form Magang

Pada gambar 3.32 merupakan bagian utama kode yang menangani proses pengajuan magang, dimulai dari validasi form yang memastikan seluruh data seperti nama, NIM, asal sekolah, jurusan, email, tanggal magang, serta dua dokumen wajib (proposal dan surat pengantar) telah diisi dengan format yang benar, termasuk memastikan file berformat PDF dan berukuran maksimal 10MB. Setelah seluruh input dinyatakan valid, sistem menyimpan kedua file tersebut ke dalam storage publik dengan nama yang telah digenerasi otomatis. Selanjutnya, data pengajuan lengkap disimpan ke tabel tb_magang menggunakan *insertGetId*, yang juga menghasilkan ID pengajuan baru untuk keperluan proses selanjutnya. Proses ini memastikan bahwa setiap pengajuan magang tersimpan dengan rapi, aman, dan dapat diproses lebih lanjut oleh admin.

3.6.9. Halaman Form Penelitian

Form pengajuan penelitian merupakan halaman yang digunakan mahasiswa untuk mengajukan permohonan penelitian ke Dinas Kesehatan. Halaman ini menyediakan formulir berisi data identitas seperti nama, NIM, asal institusi, jurusan, serta informasi terkait penelitian seperti judul, tujuan, dan waktu pelaksanaan. Selain itu, terdapat juga fitur upload dokumen pendukung seperti proposal atau surat permohonan yang harus diunggah dalam format PDF. Formulir ini dirancang agar mudah diisi dan memastikan seluruh data penting yang dibutuhkan untuk proses administratif penelitian dapat dikumpulkan secara lengkap, terstruktur, dan sesuai ketentuan sebelum diajukan ke pihak admin untuk diproses lebih lanjut.



Gambar 3. 33 Halaman Form Penelitian

Pada gambar 3.33, form pengajuan penelitian berfungsi sebagai antarmuka input yang datanya akan diproses oleh backend melalui controller penelitian. Ketika pengguna menekan tombol “Kirim”, seluruh field yang diisi dikirim ke server menggunakan HTTP POST dan melewati tahap validasi backend untuk memastikan data seperti identitas mahasiswa, detail penelitian, tanggal pelaksanaan, serta file proposal telah diisi lengkap dan sesuai format. Sistem kemudian menyimpan file proposal ke storage server dengan nama unik agar tidak terjadi duplikasi, dan seluruh data penelitian dimasukkan ke dalam tabel tb_penelitian menggunakan proses insert. Setelah penyimpanan berhasil, backend juga membuat notifikasi untuk admin agar mengetahui adanya pengajuan penelitian baru, serta mencatat riwayat aktivitas mahasiswa di tabel histori. Proses backend ini memastikan setiap pengajuan tersimpan aman, tervalidasi, dan dapat diproses lebih lanjut oleh pihak admin secara terstruktur.

3.6.10. Implementasi FormPenelitianController

```
// Validasi form dengan pesan error khusus untuk PDF
        $request->validate([
            'nama'         => 'required|string|max:255',
            'nim'          => 'required|string|max:50',
            'sekolah'      => 'required|string|max:255',
            'jurusan'      => 'required|string|max:255',
            'alamat'       => 'required|string|max:255',
            'tujuan'       => 'required|string|max:255',
            'tema'         => 'required|string|max:255',
            'mulai'        => 'required|date',
            'berakhir'     =>
'required|date|after_or_equal:mulai',
            'upload_file' =>
'required|file|mimes:pdf|max:10048',
        ], [
            // Custom error messages
            'nama.required' => 'Nama lengkap harus
diisi',
            'nim.required' => 'NIM harus diisi',
            'sekolah.required' => 'Asal
sekolah/universitas harus diisi',
            'jurusan.required' => 'Jurusan harus diisi',
            'alamat.required' => 'Alamat harus diisi',
            'tujuan.required' => 'Tujuan penelitian harus
diisi',
            'tema.required' => 'Tema penelitian harus
diisi',
            'mulai.required' => 'Tanggal mulai harus
diisi',
            'berakhir.required' => 'Tanggal berakhir
harus diisi',
            'berakhir.after_or_equal' => 'Tanggal
berakhir harus sama atau setelah tanggal mulai',
            'upload_file.required' => 'File harus
diupload',
            'upload_file.mimes' => 'File harus berformat
PDF',
            'upload_file.max' => 'Ukuran file maksimal
10MB',
        ]);

        // Simpan file
        $filePath = $request->file('upload_file')-
>storeAs(
            'uploads',
```

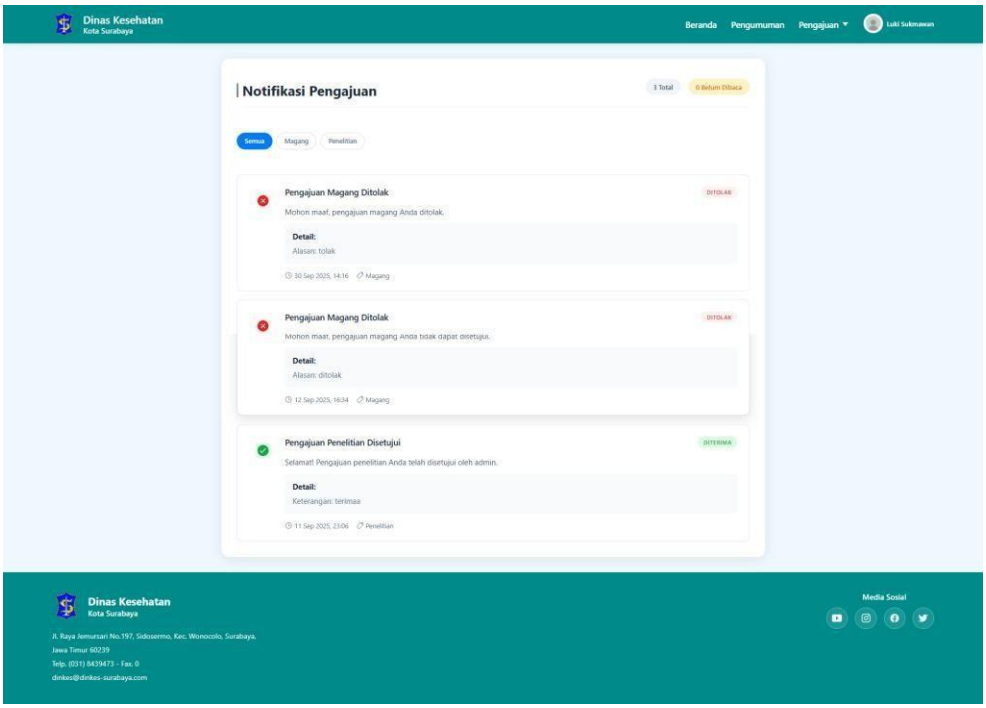
```
            time().'_penelitian.'.$request-
>file('upload_file')->getClientOriginalExtension(),
            'public'
        );

        // Simpan ke tabel tb_penelitian
        $id_penelitian = DB::table('tb_penelitian')-
>insertGetId([
            'id_user'     => $id_user,
            'nama'        => $request->nama,
            'nim'         => $request->nim,
            'sekolah'     => $request->sekolah,
            'jurusan'     => $request->jurusan,
            'alamat'      => $request->alamat,
            'tujuan'      => $request->tujuan,
            'tema'        => $request->tema,
            'mulai'       => $request->mulai,
            'berakhir'    => $request->berakhir,
            'upload_file' => basename($filePath),
            'status'      => 'pending',
        ]);
```

Gambar 3. 34 Implementasi Form Penelitian

Pada gambar 3.34 merupakan bagian utama kode yang menangani proses pengajuan penelitian, dimulai dari validasi form yang memastikan seluruh data seperti nama, NIM, asal sekolah, jurusan, alamat, tujuan penelitian, tema, tanggal penelitian, serta dokumen proposal telah diisi lengkap dan sesuai aturan, termasuk memastikan file berformat PDF dan berukuran maksimal 10MB. Setelah seluruh input dinyatakan valid, sistem menyimpan file proposal tersebut ke dalam storage publik dengan nama unik yang digenerasi otomatis. Selanjutnya, data pengajuan lengkap disimpan ke tabel tb_penelitian menggunakan insertGetId, yang sekaligus menghasilkan ID pengajuan baru untuk proses lanjutan di backend. Proses ini memastikan setiap pengajuan penelitian tersimpan dengan aman, terstruktur, dan siap diproses lebih lanjut oleh admin.

3.6.11. Halaman Notifikasi User

Sistem notifikasi mahasiswa merupakan fitur yang digunakan untuk memberikan informasi kepada user terkait status pengajuan yang mereka lakukan, baik pengajuan magang maupun penelitian. Setiap kali mahasiswa mengirimkan permohonan, sistem secara otomatis membuat notifikasi yang tersimpan pada tabel histori untuk menunjukkan bahwa pengajuan telah berhasil dikirim dan sedang menunggu proses verifikasi admin. Notifikasi ini berisi pesan singkat yang mudah dipahami serta ditampilkan pada halaman notifikasi mahasiswa, sehingga user dapat memantau perkembangan pengajuannya secara real-time. Fitur ini dirancang agar mahasiswa mendapatkan kepastian bahwa permohonan mereka telah diterima sistem, meningkatkan transparansi proses, dan memastikan komunikasi yang lebih efektif tanpa harus menunggu pemberitahuan manual dari pihak admin.



Gambar 3. 35 Halaman Notifikasi User

Pada gambar 3.35, Halaman Notifikasi Pengajuan berfungsi untuk menerima dan menampilkan notifikasi yang dikirim oleh admin sebagai hasil dari proses verifikasi pengajuan magang dan penelitian, apakah pengajuan tersebut diterima atau ditolak. Backend akan menyimpan keputusan admin ke dalam database, kemudian secara otomatis membuat data notifikasi yang terhubung dengan akun pengguna terkait. Setiap notifikasi memuat informasi jenis pengajuan, status persetujuan, keterangan dari admin, serta waktu keputusan,

3.6.12. Implementasi NotifUserController

```
{
    // 📌 Menampilkan halaman notifikasi
    public function index()
{
    $email = Session::get('email');
    $user = DB::table('tb_user')->where('email',
$email)->first();

    if (!$user) {
        return redirect()->route('login')-
>with('error_message', 'Silakan login terlebih dahulu');
    }

    $id_user = $user->id_user;

    // Ambil profil user
    $dataProfil = DB::table('tb_profil')-
>where('id_user', $id_user)->first();
    if (!$dataProfil) {
        $dataProfil = (object) [
            'nama_lengkap' => 'User',
            'nim'          => '-',
            'foto'         => ''
        ];
    }

    // ✅ Tandai semua notifikasi user sebagai sudah
dibaca
    DB::table('tb_notifMhs')
        ->where('id_user', $id_user)
        ->where('is_read', 0)
        ->update(['is_read' => 1]);
```

```
    // Ambil notifikasi setelah update
    $notifications = DB::table('tb_notifMhs as n')
        ->leftJoin('tb_magang as m', 'n.id_magang', '=',
'm.id_magang')
        ->leftJoin('tb_penelitian as p',
'n.id_penelitian', '=', 'p.id_penelitian')
        ->select(
            'n.*',
            DB::raw("CASE
                        WHEN n.id_magang IS NOT NULL
THEN m.nama
                        WHEN n.id_penelitian IS NOT NULL
THEN p.nama
                        ELSE 'Unknown'
                    END as nama_pengaju"),
            DB::raw("CASE
                        WHEN n.id_magang IS NOT NULL
THEN m.status
                        WHEN n.id_penelitian IS NOT NULL
THEN p.status
                        ELSE 'unknown'
                    END as status_pengajuan")
        )
        ->where('n.id_user', $id_user)
        ->orderBy('n.tanggal', 'DESC')
        ->get();

    // Unread count jadi 0 setelah dibuka
    $unreadCount = 0;

    return view('user.notifUser',
compact('notifications', 'unreadCount', 'dataProfil'));
}


    // 📌 Tandai notifikasi sebagai sudah dibaca
(single)
    public function markRead(Request $request, $id)
    {
        $id_user = Session::get('id_user');

        $notif = DB::table('tb_notifMhs')
            ->where('id_notifMhs', $id)
            ->where('id_user', $id_user)
            ->first();
```

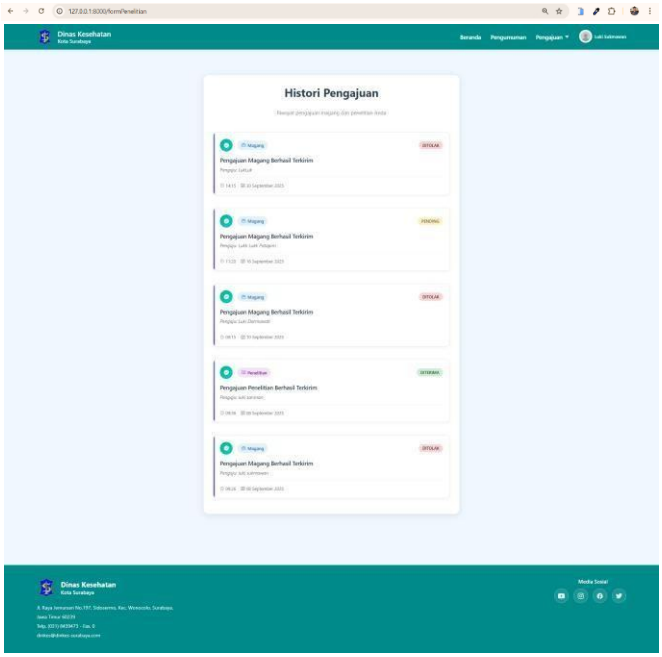
```
        if (!$notif) {
            return response()->json(['success' => false,
'error' => 'Notifikasi tidak ditemukan']);
        }

        DB::table('tb_notifMhs')
            ->where('id_notifMhs', $id)
            ->where('id_user', $id_user)
            ->update(['is_read' => 1]);

        return response()->json(['success' => true,
'message' => 'Notifikasi ditandai sudah dibaca']);
    }

    // 📌 Ambil jumlah notifikasi unread (buat AJAX
badge)
    public function getUnreadCount()
    {
        $id_user = Session::get('id_user');

        $unreadCount = DB::table('tb_notifMhs')
            ->where('id_user', $id_user)
            ->where('is_read', 0)
            ->count();

        return response()->json(['unreadCount' =>
$unreadCount]); // pakai camelCase
    }

    // 📌 Cek notifikasi baru (buat AJAX polling)
    public function checkNewNotifications()
    {
        $id_user = Session::get('id_user');

        $notifications = DB::table('tb_notifMhs')
            ->where('id_user', $id_user)
            ->where('is_read', 0)
            ->orderBy('tanggal', 'DESC')
            ->get();

        return response()->json(['notifications' =>
$notifications]);
    }
}
```

Gambar 3. 36 Implementasi Notifikasi User

Pada gambar 3.36 ini merupakan proses backend untuk fitur notifikasi pengajuan pengguna. Method index() digunakan untuk menampilkan halaman notifikasi dengan terlebih dahulu memvalidasi sesi login melalui email, mengambil data profil pengguna, kemudian menandai seluruh notifikasi sebagai sudah dibaca, serta mengambil daftar notifikasi yang terhubung dengan data magang atau penelitian lengkap dengan nama pengaju dan statusnya menggunakan *join* tabel, lalu mengirimkannya ke *view*. Method markRead() berfungsi untuk menandai satu notifikasi tertentu sebagai sudah dibaca berdasarkan ID melalui request AJAX. Method getUnreadCount() digunakan untuk mengambil jumlah notifikasi yang belum dibaca untuk ditampilkan pada badge notifikasi secara real-time. Sedangkan checkNewNotifications() digunakan untuk mengecek notifikasi baru yang belum dibaca sebagai kebutuhan *polling* AJAX agar sistem notifikasi selalu ter-*update* tanpa perlu *refresh* halaman.

3.6.13. Halaman Histori User

Pada Histori ini digunakan sebagai bentuk untuk memberikan user tersebut untuk melihat pengajuan yang sudah atau telah terkirim ke dalam web tersebut.



Gambar 3. 37 Halaman Histori User

Pada gambar 3.37, Halaman Histori Pengajuan pada sisi *backend* berfungsi untuk menampilkan riwayat seluruh pengajuan magang dan penelitian yang telah dikirim oleh pengguna ke dalam sistem. Backend mengambil data berdasarkan ID user yang sedang login dari tabel pengajuan, kemudian menampilkan informasi seperti jenis pengajuan, nama pengaju, waktu pengiriman, serta status pengajuan (pending, diterima, atau ditolak). Halaman ini digunakan sebagai sarana bagi pengguna untuk memantau seluruh aktivitas pengajuan yang telah dilakukan, memastikan bahwa data yang ditampilkan sesuai dengan riwayat yang tersimpan di database dan hanya dapat diakses oleh pengguna yang bersangkutan.

3.6.14. Implementasi HistoriUserController

```
 // Ambil histori pengajuan
        $histori = DB::table('tb_historiMhs as h')
            ->leftJoin('tb_magang as m', 'h.id_magang', '=',
'm.id_magang')
            ->leftJoin('tb_penelitian as p',
'h.id_penelitian', '=', 'p.id_penelitian')
            ->select(
                'h.*',
                'm.id_magang',
                'm.status as status_magang',
                'm.nama as nama_magang',
                'p.id_penelitian',
                'p.status as status_penelitian',
                'p.nama as nama_penelitian'
            )
            ->where('h.id_user', $id_user)
            ->where(function ($query) {
                $query->where(function ($q) {
                    $q->where('h.tipe', 'magang')
                      ->whereNotNull('m.id_magang');
                })
                ->orWhere(function ($q) {
                    $q->where('h.tipe', 'penelitian')
                      ->whereNotNull('p.id_penelitian');
                });
            })
            ->orderBy('h.tanggal', 'desc')
            ->get();

        return view('user.historiUser',
compact('dataProfil', 'unreadCount', 'histori'));
    }
}
```

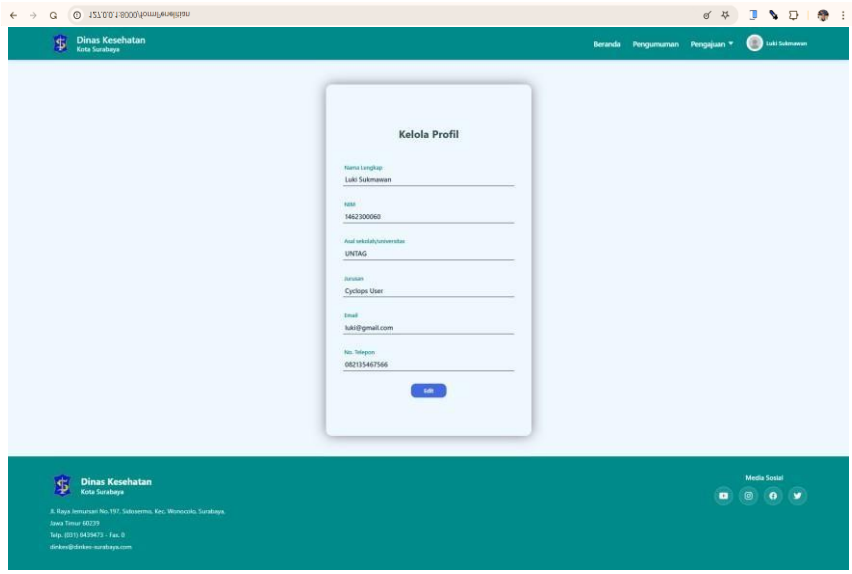
Gambar 3. 38 Implementasi Histori User

Berdasarkan Gambar 3.38, potongan kode tersebut merupakan proses *backend* untuk mengambil data histori pengajuan pengguna dari database yang tersimpan pada tabel tb_historiMhs. Data histori ini dihubungkan dengan tabel tb_magang dan tb_penelitian menggunakan *left join* untuk menampilkan detail pengajuan sesuai dengan jenisnya. Sistem hanya menampilkan data berdasarkan ID user yang sedang login.

kemudian memfilter apakah pengajuan termasuk magang atau penelitian serta memastikan datanya benar-benar ada. Selanjutnya, data diurutkan berdasarkan tanggal pengajuan terbaru, lalu dikirim ke halaman user.historiUser untuk ditampilkan kepada pengguna sebagai riwayat pengajuan yang telah dilakukan.

3.6.15. Halaman Kelola Profil

Pada Kelola profil ini digunakan sebagai mengedit akun dalam web tersebut jika memiliki kesalahan dan nama penamaan maka bisa diganti untuk nama dan lain lain.



Gambar 3. 39 Halaman Kelola Profile User

Pada gambar 3.39, merupakan Halaman Kelola Profil pada sisi *backend* berfungsi untuk mengambil, menampilkan, dan memperbarui data profil pengguna yang tersimpan di dalam database berdasarkan akun yang sedang login, seperti nama lengkap, NIM, asal sekolah/universitas, jurusan, email, dan nomor telepon. Saat halaman dibuka, backend melakukan proses query data profil berdasarkan id_user, kemudian mengirimkannya ke *view* untuk ditampilkan pada form.

Ketika pengguna menekan tombol Simpan, backend akan melakukan validasi data, lalu memperbarui informasi tersebut ke tabel profil agar perubahan langsung tersimpan dan dapat digunakan pada seluruh fitur sistem.

3.6.16. Implementasi UserProfilController

```
    public function store(Request $request)
    {
        // Cek apakah user sudah login
        if (!Session::has('email')) {
            return redirect()->route('login')-
>with('error', 'Silakan login terlebih dahulu');
        }

        $email = Session::get('email');
        $role = Session::get('role');

        // Cek role user
        if ($role !== 'user') {
            return redirect()->route('login')-
>with('error', 'Akses ditolak');
        }

        // Ambil data user
        $dataUser = DB::table('tb_user')
            ->select('id_user')
            ->where('email', $email)
            ->first();

        if (!$dataUser) {
            Session::flush();
            return redirect()->route('login')-
>with('error', 'User tidak ditemukan');
        }

        $id_user = $dataUser->id_user;

        // Validasi input
        $validated = $request->validate([
            'nama'     => 'required|string|max:255',
            'nim'      => 'required|string|max:50',
            'sekolah'  => 'required|string|max:255',
            'jurusan'  => 'required|string|max:255',
            'email'    => 'required|email|max:255',
```

```
            'telepon'  => 'required|string|max:20',
        ], [
            'nama.required' => 'Nama lengkap wajib
diisi',
            'nim.required' => 'NIM wajib diisi',
            'sekolah.required' => 'Asal sekolah wajib
diisi',
            'jurusan.required' => 'Jurusan wajib diisi',
            'email.required' => 'Email wajib diisi',
            'email.email' => 'Format email tidak valid',
            'telepon.required' => 'No. telepon wajib
diisi',
        ]);

        try {
            // Cek apakah sudah ada profil
            $existingProfil = DB::table('tb_profil')
                ->where('id_user', $id_user)
                ->first();

            if ($existingProfil) {
                // Update profil (sesuai kolom tb_profil)
                DB::table('tb_profil')
                    ->where('id_user', $id_user)
                    ->update([
                        'nama_lengkap' =>
$validated['nama'],
                        'nim'          =>
$validated['nim'],
                        'asal_sekolah' =>
$validated['sekolah'],
                        'jurusan'      =>
$validated['jurusan'],
                        'email'        =>
$validated['email'],
                        'no_telepon'   =>
$validated['telepon'],
                    ]);

                $message = "Profil berhasil diperbarui!";
            } else {
                // Insert profil baru (sesuai kolom
tb_profil)
                DB::table('tb_profil')->insert([
                    'id_user'      => $id_user,
                    'nama_lengkap' => $validated['nama'],
                    'nim'          => $validated['nim'],
```

```
                    'asal_sekolah' =>
$validated['sekolah'],
                    'jurusan'      =>
$validated['jurusan'],
                    'email'        =>
$validated['email'],
                    'no_telepon'   =>
$validated['telepon'],
                ]);

                $message = "Profil berhasil disimpan!";
            }

            // Jika profil sebelumnya forceComplete,
arahkan ke beranda
            if ($request->has('force')) {
                return redirect()->route('beranda')-
>with('success', $message);
            }

            return redirect()->back()->with('success',
$message);

        } catch (\Exception $e) {
            return redirect()->back()
                ->with('error', 'Terjadi kesalahan saat
menyimpan profil: ' . $e->getMessage())
                ->withInput();
        }
    }
}
```

Gambar 3. 40 Implementasi User profil

Pada gambar 3.40, Kode store() ini berfungsi sebagai proses *backend* untuk menyimpan dan memperbarui data profil pengguna, sekaligus digunakan saat pengguna baru pertama kali login dan diwajibkan mengisi profil terlebih dahulu. Alur dimulai dengan pengecekan apakah user sudah login dan memiliki role *user*, kemudian sistem mengambil id_user berdasarkan email yang tersimpan di session.

Setelah itu, seluruh input profil divalidasi agar sesuai format yang ditentukan. Jika pengguna sudah memiliki data profil, maka sistem akan melakukan proses update data; jika belum memiliki profil, maka sistem akan melakukan insert data baru ke tabel tb_profil. Proses ini memastikan bahwa baik pengguna lama yang mengubah data maupun pengguna baru yang baru pertama masuk tetap menggunakan satu mekanisme penyimpanan profil yang sama, lalu sistem akan mengarahkan kembali ke halaman sebelumnya atau ke beranda setelah profil berhasil disimpan.

3.6.17. Halaman Daftar Magang Admin

Gambar 3. 41 Halaman Daftar Magang pada Admin

Pada gambar 3.41, Halaman Daftar Magang Admin adalah halaman yang menampilkan seluruh pengajuan magang yang telah diajukan oleh mahasiswa dalam bentuk tabel. Halaman ini berfungsi sebagai pusat monitoring dan manajemen pengajuan magang oleh admin. Admin dapat Melihat semua data pengajuan dan memantau status pengajuan.

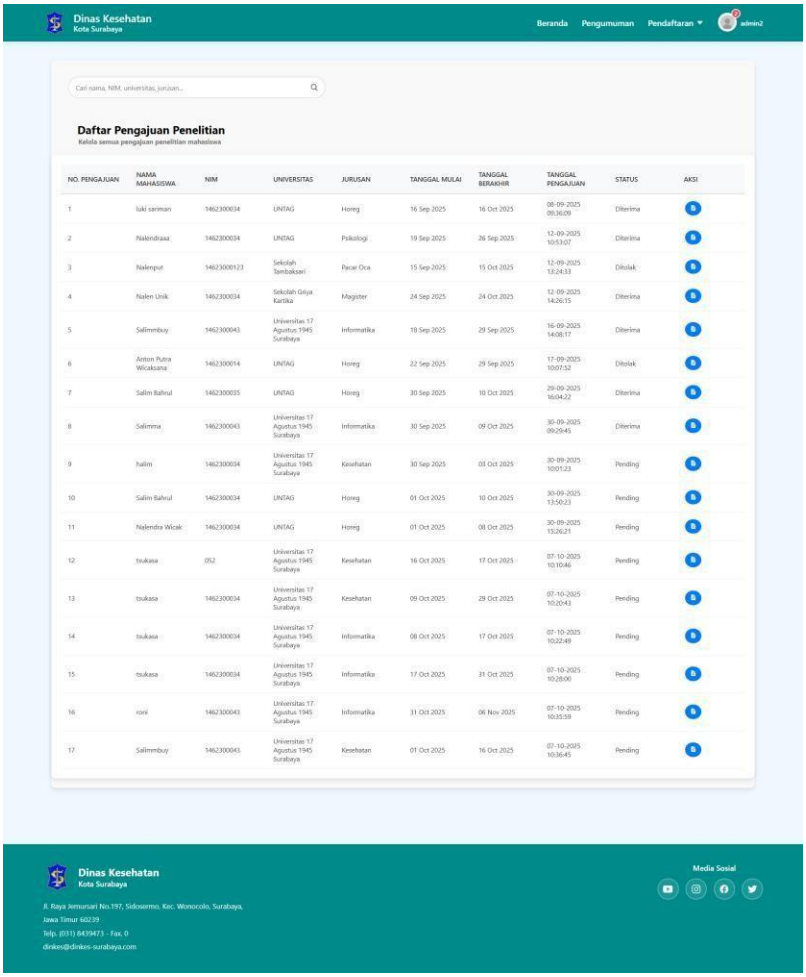
3.6.18. Implementasi DaftarMagangController

```
public function index()
    {
        // Ambil email dan role dari session
        $email = session('email');
        $role = session('role');

        if (!$email || $role !== 'admin') {
            return redirect('/login'); // Redirect ke login jika
bukan admin
        }

        // Ambil data user admin
        $dataUser = DB::table('tb_user')
            ->select('id_user', 'username', 'email','role')
            ->where('email', $email)
            ->first();

        // Hitung notifikasi yang belum dibaca
        $unread_count = DB::table('tb_notifAdmin')
            ->where('is_read', 0)
            ->count();

        // Ambil semua data magang
        $pengajuanMagang = DB::table('tb_magang')->get();

        // Kirim data ke view Blade
        return view('admin.daftarMagang', [
            'dataUser' => $dataUser,
            'unread_count' => $unread_count,
            'pengajuanMagang' => $pengajuanMagang,
            'email' => $email,
        ]);
    }
```

Gambar 3. 42 Implementasi Daftar Magang pada Admin

Pada gambar 3.42, Method index() melakukan validasi admin, mengambil data user dan notifikasi, kemudian query db table (tb magang) mengambil seluruh pengajuan magang beserta field status (menunggu/diterima/ditolak), alasan (jika ditolak), dan keterangan (jika diterima) untuk ditampilkan dalam tabel di view Blade dengan badge berwarna sesuai status masing-masing.

3.6.19. Halaman Daftar Penelitian Admin



Gambar 3. 43 Halaman Daftar Penelitian

Pada gambar 3.43, Halaman Daftar Penelitian Admin adalah halaman yang menampilkan seluruh pengajuan penelitian yang telah diajukan oleh mahasiswa dalam bentuk tabel. Halaman ini berfungsi sebagai pusat monitoring dan manajemen pengajuan magang oleh admin. Admin dapat Melihat semua data pengajuan dan memantau status pengajuan.

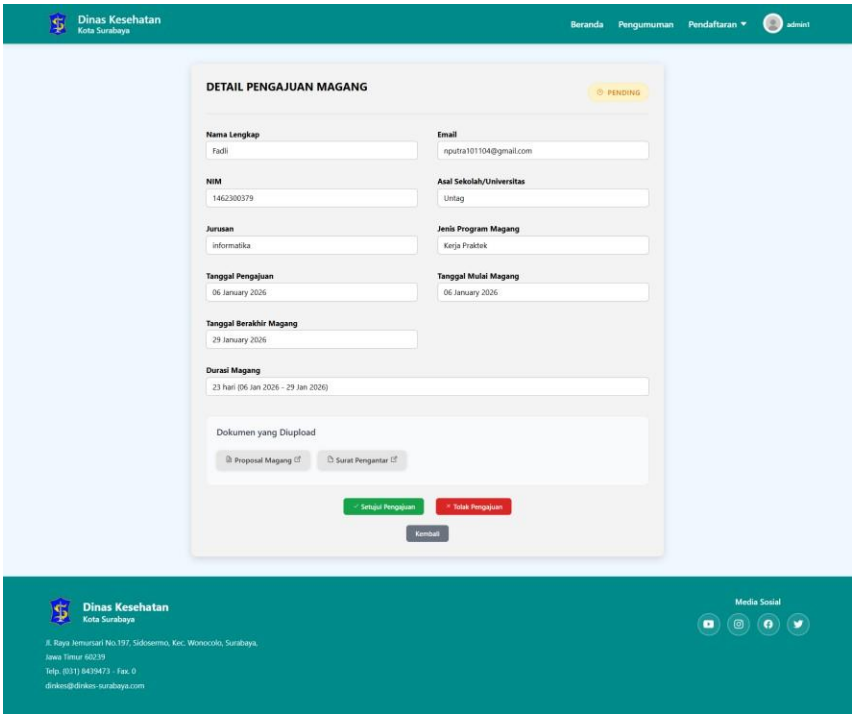
3.6.20. Implementasi DaftarPenelitianController

```
public function index()
    {
        // Ambil email dan role dari session
        $email = session('email');
        $role = session('role');

        if (!$email || $role !== 'admin') {
            return redirect('/login'); // Redirect ke login
jika bukan admin
        }

        // Ambil data user admin
        $dataUser = DB::table('tb_user')
            ->select('id_user', 'username', 'email', 'role')
            ->where('email', $email)
            ->first();

        // Hitung notifikasi yang belum dibaca
        $unread_count = DB::table('tb_notifAdmin')
            ->where('is_read', 0)
            ->count();

        // Ambil semua pengajuan penelitian
        $pengajuanPenelitian = DB::table('tb_penelitian')-
>get();

        // Kirim data ke Blade
        return view('admin.daftarPenelitian', [
            'dataUser' => $dataUser,
            'unread_count' => $unread_count,
            'pengajuanPenelitian' => $pengajuanPenelitian,
            'email' => $email,
        ]);
    }
```

Gambar 3. 44 Implementasi Daftar Penelitian

Pada gambar 3.44, Method index() melakukan validasi admin, mengambil data user dan notifikasi, kemudian query db table (tb penelitian) mengambil seluruh pengajuan penelitian beserta field status (menunggu/diterima/ditolak), alasan (jika ditolak), dan keterangan (jika diterima) untuk ditampilkan dalam 85able di view Blade dengan badge berwarna sesuai status masing-masing.

3.6.21. Halaman Detail Magang Admin



Gambar 3. 45 Halaman Detail Magang

Pada gambar 3.45, Halaman Detail Magang Admin merupakan halaman yang menampilkan informasi lengkap dari satu pengajuan penelitian yang dipilih oleh admin dari halaman daftar magang, yang paling penting adalah field status yang menampilkan kondisi terkini pengajuan (menunggu/diterima/ditolak/draft/selesai) dengan badge berwarna di pojok kanan atas halaman. Jika status pengajuan masih "menunggu", halaman ini menyediakan dua tombol aksi yaitu tombol hijau "Setuju Pengajuan" yang akan membuka modal konfirmasi untuk menyetujui pengajuan dengan opsi menambahkan keterangan tambahan, dan tombol merah "Tolak Pengajuan" yang akan membuka modal untuk menolak pengajuan dengan wajib mengisi alasan penolakan.

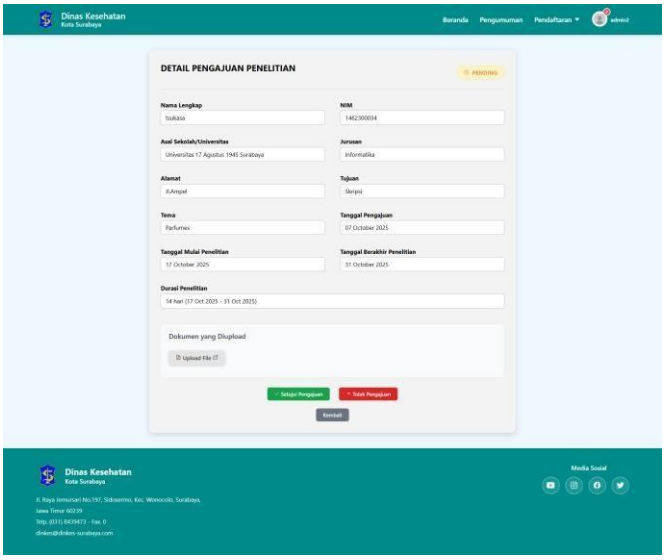
3.6.22. Implementasi MagangController

```
public function show(Request $request, $id_magang)
    {
        // Cek session role admin
        // Ambil email dan role dari session
        $email = session('email');
        $role = session('role');

        if (!$email || $role !== 'admin') {
            return redirect('/login'); // Redirect ke login jika
bukan admin
        }

        // Ambil data user admin
        $dataUser = DB::table('tb_user')
            ->select('id_user', 'username', 'email','role')
            ->where('email', $email)
            ->first();

        // Hitung notifikasi yang belum dibaca
        $unread_count = DB::table('tb_notifAdmin')
            ->where('is_read', 0)
            ->count();

        // Ambil detail magang
        $data = DB::table('tb_magang')
            ->where('id_magang', intval($id_magang))
            ->first();

        if (!$data) {
            return redirect()->route('daftar.magang');
        }

        // Kirim ke view
        return view('admin.detailMagang', [
            'dataUser' => $dataUser,
            'unread_count' => $unread_count,
            'data' => $data
        ]);
    }
```

Gambar 3. 46 Implementasi Detail Magang Admin

Pada gambar 3.46 Method show() melakukan validasi admin, mengambil data user dan notifikasi, kemudian query tb magang dan id magang mengambil satu record detail magang berdasarkan ID yang dikirim melalui URL parameter, menghasilkan object yang berisi seluruh informasi magang termasuk field status untuk menampilkan badge status di pojok kanan (menunggu/diterima/ditolak), field alasan untuk menampilkan alasan jika pengajuan ditolak, field keterangan untuk menampilkan catatan tambahan jika pengajuan diterima, field proposal dan pengantar yang berisi nama file dokumen yang dapat dilihat atau diunduh melalui method viewFile() di MagangController dengan parameter type ('proposal' atau 'pengantar') yang akan memanggil Storage facade untuk mengakses file dari direktori storage/app/public/uploads/, dimana jika data tidak ditemukan sistem akan redirect kembali ke halaman daftar magang, dan jika status masih "menunggu" maka halaman akan menampilkan tombol "Setuju Pengajuan" dan "Tolak Pengajuan" yang akan memanggil method updateStatus() di MagangController untuk memproses persetujuan atau penolakan. Jika sudah status sudah diterima maupun ditolak maka akan kembali dan hanya ada tombol kembali.

3.6.23. Halaman Detail Penelitian Admin



Gambar 3. 47 Halaman Detail Penelitian

Pada gambar 3.47, Halaman Detail Penelitian Admin merupakan halaman yang menampilkan informasi lengkap dari satu pengajuan penelitian yang dipilih oleh admin dari halaman daftar penelitian, yang paling penting adalah field status yang menampilkan kondisi terkini pengajuan (menunggu/diterima/ditolak/draft/selesai) dengan badge berwarna di pojok kanan atas halaman. Jika status pengajuan masih "menunggu", halaman ini menyediakan dua tombol aksi yaitu tombol hijau "Setuju Pengajuan" yang akan membuka modal konfirmasi untuk menyetujui pengajuan dengan opsi menambahkan keterangan tambahan, dan tombol merah "Tolak Pengajuan" yang akan membuka modal untuk menolak pengajuan dengan wajib mengisi alasan penolakan.

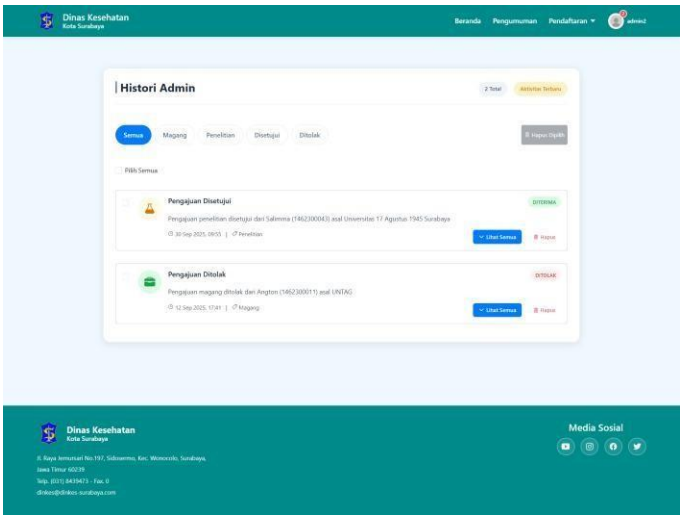
3.6.24. Implementasi PenelitianController

```
public function show(Request $request, $id_penelitian)
    {
        // Cek session role admin
        // Ambil email dan role dari session
        $email = session('email');
        $role = session('role');

        if (!$email || $role !== 'admin') {
            return redirect('/login'); // Redirect ke login
jika bukan admin
        }

        // Ambil data user admin
        $dataUser = DB::table('tb_user')
            ->select('id_user', 'username', 'email','role')
            ->where('email', $email)
            ->first();

        // Hitung notifikasi yang belum dibaca
        $unread_count = DB::table('tb_notifAdmin')
            ->where('is_read', 0)
            ->count();

        // Ambil detail penelitian
        $data = DB::table('tb_penelitian')
            ->where('id_penelitian', intval($id_penelitian))
            ->first();

        if (!$data) {
            return redirect()->route('daftar.penelitian');
        }

        // Kirim ke view
        return view('admin.detailPenelitian', [
            'dataUser' => $dataUser,
            'unread_count' => $unread_count,
            'data' => $data
        ]);
    }
```

Gambar 3. 48 Implementasi Detail Penelitian Admin

Pada gambar 3.48 Method show() melakukan validasi admin, mengambil data user dan notifikasi, kemudian query tb penelitian dan id penelitian mengambil satu record detail penelitian berdasarkan ID yang dikirim melalui URL parameter, menghasilkan object yang berisi seluruh informasi magang termasuk field status untuk menampilkan badge status di pojok kanan (menunggu/diterima/ditolak), field alasan untuk menampilkan alasan jika pengajuan ditolak, field keterangan untuk menampilkan catatan tambahan jika pengajuan diterima, field proposal dan pengantar yang berisi nama file dokumen yang dapat dilihat atau diunduh melalui method viewFile() di PenelitianController dengan parameter type ('proposal penelitian') yang akan memanggil Storage facade untuk mengakses file dari direktori storage/app/public/uploads/, dimana jika data tidak ditemukan sistem akan redirect kembali ke halaman daftar magang, dan jika status masih "menunggu" maka halaman akan menampilkan tombol "Setuju Pengajuan" dan "Tolak Pengajuan" yang akan memanggil method updateStatus() di PenelitianController untuk memproses persetujuan atau penolakan. Jika sudah status sudah diterima maupun ditolak maka akan kembali dan hanya ada tombol kembali.

3.6.25. Halaman Histori Admin



Gambar 3. 49 Halaman Histori Admin

Pada gambar 3.49 adalah halaman histori admin merupakan halaman yang menampilkan catatan jejak aktivitas semua admin dalam mengelola pengajuan magang dan penelitian sebagai fitur audit trail atau log aktivitas sistem. Pada halaman ini digunakan sebagai jejak aktivitas untuk memberikan bahwa user tersebut telah diterima maupun ditolak.

3.6.26. Implementasi HistoriAdminController

```
public function index(Request $request)
    {
        $email = session('email');
        $role = session('role');

        // Cek login dan role
        if (!$email || $role !== 'admin') {
            return redirect('login');
        }

        // Ambil data user
        $dataUser = DB::table('tb_user')
            ->select('id_user', 'username', 'email')
            ->where('email', $email)
            ->first();

        if (!$dataUser) {
            return redirect('login');
        }

        $id_user = $dataUser->id_user;

        // Hitung notifikasi unread
        $unread_count = DB::table('tb_notifAdmin')
            ->where('is_read', 0)
            ->count();

        // Ambil data histori join dengan magang / penelitian
        $historiData = DB::table('tb_historiAdmin as ha')
            ->leftJoin('tb_magang as m', 'ha.id_magang', '=',
'm.id_magang')
            ->leftJoin('tb_penelitian as p', 'ha.id_penelitian',
'=', 'p.id_penelitian')
            ->select(
                'ha.*',
                DB::raw("CASE
```

```
                    WHEN ha.id_magang IS NOT NULL THEN m.nama
                    WHEN ha.id_penelitian IS NOT NULL THEN p.nama
                    ELSE 'Unknown'
                END as nama_pengaju"),
                DB::raw("CASE
                    WHEN ha.id_magang IS NOT NULL THEN m.sekolah
                    WHEN ha.id_penelitian IS NOT NULL THEN
p.sekolah
                    ELSE 'Unknown'
                END as sekolah_pengaju"),
                DB::raw("CASE
                    WHEN ha.id_magang IS NOT NULL THEN m.nim
                    WHEN ha.id_penelitian IS NOT NULL THEN p.nim
                    ELSE 'Unknown'
                END as nim_pengaju"),
                DB::raw("CASE
                    WHEN ha.id_magang IS NOT NULL THEN m.jurusan
                    WHEN ha.id_penelitian IS NOT NULL THEN
p.jurusan
                    ELSE 'Unknown'
                END as jurusan_pengaju"),
                DB::raw("CASE
                    WHEN ha.id_magang IS NOT NULL THEN m.alasan
                    WHEN ha.id_penelitian IS NOT NULL THEN
p.alasan
                    ELSE NULL
                END as alasan_pengaju"),
                DB::raw("CASE
                    WHEN ha.id_magang IS NOT NULL THEN
m.keterangan
                    WHEN ha.id_penelitian IS NOT NULL THEN
p.keterangan
                    ELSE NULL
                END as keterangan_admin")
            )
            ->where('ha.id_user', $id_user)
            ->orderBy('ha.tanggal', 'desc')
            ->get();

        return view('admin.historiAdmin', [
            'dataUser' => $dataUser,
            'email' => $dataUser->email,
            'unread_count' => $unread_count,
            'historiData' => $historiData,
        ]);
    }
```

Gambar 3. 50 Implementasi Histori Admin

Pada gambar 3.50, Method index() melakukan validasi admin dan mengambil id_user admin yang sedang login, kemudian melakukan query kompleks dengan LEFT JOIN antara tb_historiAdmin dengan tb_magang dan tb_penelitian menggunakan CASE statement SQL untuk mengambil data pengaju dari tabel yang sesuai, jika id_magang terisi maka ambil dari tb_magang dan jika id_penelitian terisi maka ambil dari tb_penelitian, menghasilkan field tambahan nama_pengaju, sekolah_pengaju, nim_pengaju, jurusan_pengaju, alasan_pengaju untuk alasan penolakan, dan keterangan_admin untuk keterangan persetujuan, dengan filter where ha.id_user sama dengan id_user untuk hanya menampilkan histori milik admin yang sedang login bersifat personal bukan histori semua admin, diurutkan dari tanggal terbaru menggunakan orderBy ha.tanggal desc,

```
public function delete($id)
    {
        $role = session('role');
        if ($role !== 'admin') {
            return response()->json(['success' => false,
'error' => 'Unauthorized']);
        }

        $deleted = DB::table('tb_historiAdmin')-
>where('id_historiAdmin', $id)->delete();

        if ($deleted) {
            return response()->json(['success' => true,
'message' => 'Histori berhasil dihapus']);
        } else {
            return response()->json(['success' => false,
'error' => 'Failed to delete histori']);
        }
    }
```

Gambar 3. 51 Implementasi Delete pada Histori Admin

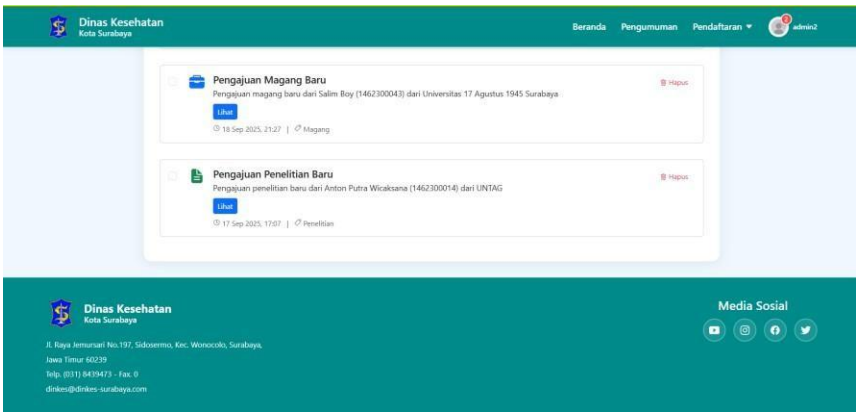
Pada gambar 3.51 merupakan delete dengan satu tujuan, yaitu menghapus satu record histori berdasarkan id_historiAdmin

```
    public function bulkDelete(Request $request)
    {
        $role = session('role');
        if ($role !== 'admin') {
            return response()->json(['success' => false,
'error' => 'Unauthorized']);
        }

        $ids = $request->input('ids');
        if (!is_array($ids) || empty($ids)) {
            return response()->json(['success' => false,
'error' => 'Invalid history IDs']);
        }

        $deleted = DB::table('tb_historiAdmin')-
>whereIn('id_historiAdmin', $ids)->delete();

        if ($deleted) {
            return response()->json(['success' => true,
'message' => count($ids) . ' histori berhasil dihapus']);
        } else {
            return response()->json(['success' => false,
'error' => 'Failed to delete history records']);
        }
    }
```

Gambar 3. 52 Implementasi Delete Semua Histori

Pada gambar 3.52 merupakan delete dengan tujuan yaitu menghapus semua record histori berdasarkan id_historiAdmin

3.6.27. Halaman Notifikasi Admin



Gambar 3. 53 Halaman Notifikasi Admin

Pada gambar 3.53, Halaman Notifikasi Admin merupakan halaman yang menampilkan daftar pemberitahuan tentang pengajuan magang dan penelitian baru yang masuk ke sistem. Setiap notifikasi menampilkan informasi berupa judul pengajuan, nama lengkap pengaju beserta NIM, asal sekolah atau universitas, tipe pengajuan (magang atau penelitian), dan tanggal waktu pengajuan dibuat. Admin dapat mengklik tombol "Lihat" pada setiap notifikasi untuk membuka halaman detail pengajuan yang sesuai, baik detail magang maupun detail penelitian, dimana admin dapat melakukan persetujuan atau penolakan. Setiap notifikasi dilengkapi dengan checkbox di sebelah kiri untuk memilih notifikasi yang akan dihapus secara massal, dan tombol "Hapus" di pojok kanan atas setiap card notifikasi untuk menghapus notifikasi individual, serta sistem akan menampilkan badge atau indikator visual untuk membedakan notifikasi yang sudah dibaca dan belum dibaca agar admin dapat fokus pada pengajuan baru yang masih memerlukan tindakan.

3.6.28. Implementasi NotifAdminController

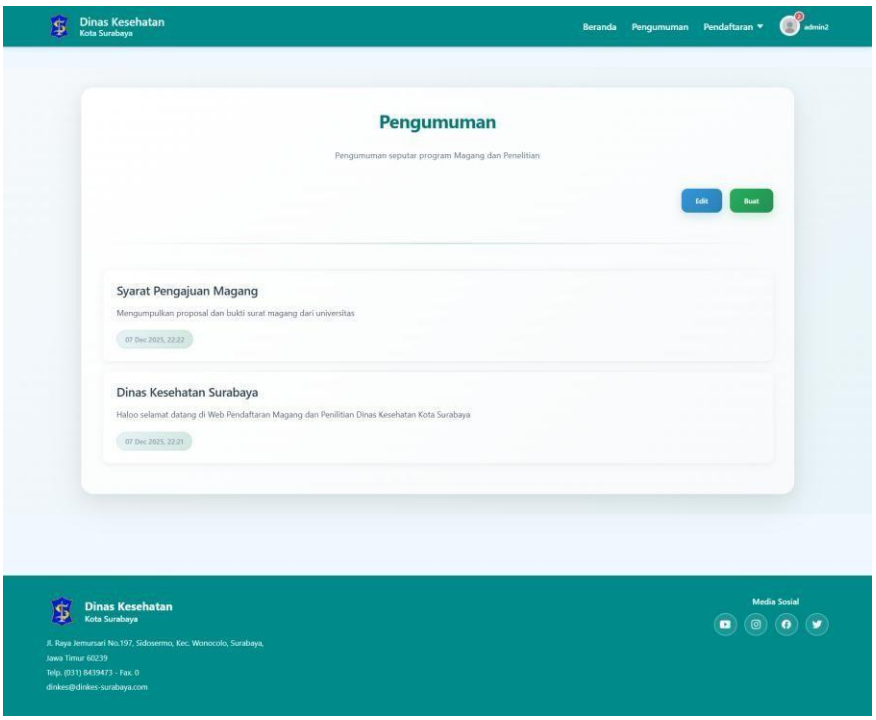
```
public function index(Request $request)
    {
        $email = session('email');
        $role = session('role');

        // Cek login dan role
        if (!$email || $role !== 'admin') {
            return redirect('login');
        }

        // Ambil data admin dari tb_user
        $dataUser = DB::table('tb_user')
            ->select('id_user', 'username', 'email')
            ->where('email', $email)
            ->where('role', 'admin')
            ->first();

        if (!$dataUser) {
            return redirect('login');
        }

        $id_user = $dataUser->id_user;

        // Hitung notifikasi unread
        $unread_count = DB::table('tb_notifAdmin')
            ->where('is_read', 0)
            ->count();

        // Ambil semua notifikasi
        $notifications = DB::table('tb_notifAdmin as na')
            ->leftJoin('tb_user as u', 'na.id_user', '=',
'u.id_user')
            ->leftJoin('tb_magang as m', 'na.id_magang', '=',
'm.id_magang')
            ->leftJoin('tb_penelitian as p', 'na.id_penelitian',
'=', 'p.id_penelitian')
            ->select(
                'na.*',
                'u.email',
                DB::raw("CASE
                    WHEN na.id_magang IS NOT NULL THEN m.nama
                    WHEN na.id_penelitian IS NOT NULL THEN p.nama
                    ELSE 'Unknown'
                END as nama_pengaju"),
```

```
            DB::raw("CASE
                WHEN na.id_magang IS NOT NULL THEN m.sekolah
                WHEN na.id_penelitian IS NOT NULL THEN p.sekolah
                ELSE 'Unknown'
            END as sekolah_pengaju"),
            DB::raw("CASE
                WHEN na.id_magang IS NOT NULL THEN m.status
                WHEN na.id_penelitian IS NOT NULL THEN p.status
                ELSE 'pending'
            END as status_pengajuan")
        )
        ->orderBy('na.tanggal', 'desc')
        ->get();

    return view('admin.notifAdmin', [
        'dataUser'     => $dataUser,
        'email'        => $dataUser->email,
        'unread_count' => $unread_count,
        'notifications'=> $notifications,
    ]);
}
```

Gambar 3. 54 Implementasi Notifikasi Admin

Pada gambar 3.54, Method index() mengambil notifikasi dari tb_notifAdmin dengan LEFT JOIN ke tb_magang dan tb_penelitian menggunakan case statement untuk mendapatkan nama_pengaju sekolah_pengaju dan status_pengajuan diurutkan tanggal terbaru, method markRead() untuk tandai dibaca dengan update is_read menjadi 1, lalu pada tipe delete sama seperti pada implementasi notifikasi admin gambar 3.49 dan gambar 3.50.

3.6.29. Halaman Pengumuman Admin



Gambar 3. 55 Halaman Pengumuman Admin

Pada gambar 3.55, Halaman Pengumuman Admin merupakan halaman yang menampilkan daftar pengumuman yang telah dibuat oleh admin dan akan ditampilkan kepada mahasiswa pengguna sistem. Setiapengumuman menampilkan informasi berupa judul pengumuman, isi atau konten pengumuman, dan tanggal waktu pengumuman dibuat atau diperbarui. Halaman ini menyediakan tombol "Tambah Pengumuman Baru" di bagian atas untuk membuka modal form pengisian judul dan isi pengumuman , Setiap card pengumuman dilengkapi dengan tombol "Edit" untuk membuka modal form edit yang memungkinkan admin mengubah judul dan isi pengumuman yang sudah ada ."Hapus" untuk menghapus pengumuman dari system. Pada semua pengumuman diurutkan berdasarkan tanggal terbaru sehingga pengumuman yang baru ditambahkan atau diperbarui akan muncul di bagian atas daftar untuk memudahkan admin dalam mengelola informasi terkini yang perlu disampaikan kepada mahasiswa

3.6.30. Implementasi PengumumanAdminController

```
public function index()
    {
        // cek session & role admin
        if (!Session::has('email') || Session::get('role') !==
'admin') {
            return redirect()->route('login');
        }

        $email = Session::get('email');

        // ambil user info
        $dataUser = DB::table('tb_user')
            ->select('id_user', 'username', 'email')
            ->where('email', $email)
            ->first();

        // hitung notifikasi yang belum dibaca
        $unread_count = DB::table('tb_notifAdmin')
            ->where('is_read', 0)
            ->count();

        // ambil semua pengumuman
        $pengumuman = DB::table('tb_pengumuman')
            ->orderBy('tanggal', 'DESC')
            ->get();

        return view('admin.pengumumanAdmin', [
            'dataUser' => $dataUser,
            'email' => $email,
            'unread_count' => $unread_count,
            'pengumuman' => $pengumuman
        ]);
    }
```

Gambar 3. 56 Implementasi Pengumuman Admin

Pada gambar 3.56, Method index() mengambil semua pengumuman dari tb_pengumuman diurutkan tanggal DESC untuk menampilkan terbaru di atas, method store() untuk tambah pengumuman dengan generate ID manual mengambil max id_pengum tambah 1 kemudian insert judul isi dan tanggal, method update() untuk edit pengumuman dengan validasi keberadaan data. Lalu ada juga tahapan Tambah pengumuman pada

gambar 3.57, update pengumuman pada gambar 3.58, dan hapus pengumuman pada gambar 3.59.

```
public function store(Request $request)
    {
        if (!Session::has('email') || Session::get('role') !== 
'admin') {
            return redirect()->route('login');
        }

        $request->validate([
            'judul' => 'required|string',
            'isi'   => 'required|string',
        ]);

        // Generate ID manual (ambil ID terbesar + 1)
        $maxId = DB::table('tb_pengumuman')->max('id_pengum');
        $newId = $maxId ? $maxId + 1 : 1;

        $insert = DB::table('tb_pengumuman')->insert([
            'id_pengum' => $newId,
            'judul'     => $request->input('judul'),
            'isi'       => $request->input('isi'),
            'tanggal'   => now(),
        ]);

        if ($insert) {
            return redirect()->back()->with('success', 
'Pengumuman berhasil ditambahkan!');
        } else {
            return redirect()->back()->with('error', 'Gagal 
menambahkan pengumuman!');
        }
    }
```

Gambar 3. 57 Implementasi Tambah Pengumuman

```
public function update(Request $request, $id)
{
    if (!Session::has('email') || Session::get('role') !==
'admin') {
        return redirect()->route('login');
    }

    $request->validate([
        'judul' => 'required|string',
        'isi'   => 'required|string',
    ]);

    // Cek apakah pengumuman ada
    $pengumuman = DB::table('tb_pengumuman')
        ->where('id_pengum', $id)
        ->first();

    if (!$pengumuman) {
        return redirect()->back()->with('error', 'Pengumuman
tidak ditemukan!');
    }

    $update = DB::table('tb_pengumuman')
        ->where('id_pengum', $id)
        ->update([
            'judul'   => $request->input('judul'),
            'isi'     => $request->input('isi'),
            'tanggal' => now(), // update timestamp juga
        ]);

    return redirect()->route('admin.pengumumanAdmin')-
>with('success', 'Pengumuman berhasil diupdate!');
}
```

Gambar 3. 58 Update pengumuman

```
 public function destroy($id)
    {
        if (!Session::has('email') || Session::get('role') !==
'admin') {
            return redirect()->route('login');
        }

        $delete = DB::table('tb_pengumuman')
            ->where('id_pengum', $id)
            ->delete();

        if ($delete) {
            return redirect()->back()->with('success',
'Pengumuman berhasil dihapus!');
        } else {
            return redirect()->back()->with('error', 'Gagal
menghapus pengumuman!');
        }
    }
                                    }
```

Gambar 3. 59 Hapus Pengumuman

BAB 4
KESIMPULAN DAN SARAN

4.1. Kesimpulan

Berdasarkan pelaksanaan kerja praktek yang telah dilakukan di Dinas Kesehatan Kota Surabaya, dapat diambil beberapa kesimpulan sebagai berikut::

1. Sistem backend pendaftaran magang dan penelitian berbasis web telah berhasil dirancang dan dibangun menggunakan framework Laravel dengan arsitektur MVC yang terintegrasi dengan database
2. Implementasi sistem berhasil mencakup fungsi manajemen data pendaftar, verifikasi pengajuan, serta sistem notifikasi status pengajuan secara real-time yang memudahkan komunikasi antara mahasiswa dan admin.
3. Struktur database relasional dengan 18 tabel yang saling berelasi telah berhasil diimplementasikan untuk mengelola data user, profil, pengajuan magang dan penelitian, notifikasi, histori aktivitas, serta pengumuman secara efisien dan terorganisir.
4. Sistem berhasil meningkatkan efisiensi dan akurasi dalam pengelolaan data pendaftaran dengan mengurangi proses manual, mempercepat verifikasi dokumen, dan memberikan transparansi status pengajuan kepada mahasiswa.
5. Pelaksanaan kerja praktek ini memberikan pengalaman berharga dalam pengembangan sistem informasi berbasis web, meningkatkan kemampuan teknis dalam implementasi backend, serta mengembangkan kemampuan komunikasi, kolaborasi, dan manajemen waktu dalam lingkungan kerja profesional.

4.2. Saran

Berdasarkan pengalaman selama pelaksanaan kerja praktek, beberapa saran yang dapat diberikan adalah:

1. Sistem dapat dikembangkan lebih lanjut dengan menambahkan fitur dashboard analytics untuk memantau statistik pengajuan secara komprehensif dan membantu pengambilan keputusan
2. Perlu dilakukan pengembangan fitur notifikasi melalui WhatsApp atau SMS sebagai alternatif komunikasi selain email untuk meningkatkan keterjangkauan informasi kepada mahasiswa.
3. Sistem dapat ditingkatkan dengan menambahkan fitur penjadwalan otomatis untuk mengatur jadwal magang dan penelitian berdasarkan kuota dan periode yang tersedia.
4. Disarankan untuk melakukan pelatihan penggunaan sistem secara berkala kepada admin dan staf Dinas Kesehatan agar dapat memaksimalkan pemanfaatan seluruh fitur yang tersedia.
5. Perlu dilakukan backup database secara rutin dan implementasi disaster recovery plan untuk menjaga keamanan dan ketersediaan data sistem.

DAFTAR PUSTAKA

Dennis, A., Wixom, B.H. dan Tegarden, D., 2015. *Systems Analysis and Design: An Object-Oriented Approach with UML*. New Jersey: John Wiley & Sons.

Doroodchi, M. dan Dastgheib, S., 2008. *A Comparative Study of Web Development Technologies Using Open Source and Proprietary Software*. International Journal of Web & Semantic Technology.

Kurniawan, R. dan Romzi, M., 2022. *Analisis Perbandingan Editor Kode untuk Pengembangan Aplikasi Web*. Jurnal Teknologi Informasi dan Komputer.

Permatasari, D. dan Suhendi, A., 2020. *Pengembangan Website Responsif Menggunakan HTML5 dan CSS3*. Jurnal Sistem Informasi dan Teknologi.

Permatasari, D. dan Suhendi, A., 2020. *Pengembangan Website Responsif Menggunakan HTML5 dan CSS3*. Jurnal Sistem Informasi dan Teknologi

Putra, A., Ferdinandus, E. dan Bayu, S., 2019. *Perancangan Basis Data Menggunakan Entity Relationship Diagram pada Sistem Informasi*. Jurnal Informatika dan Sistem Informasi.

Rosaly, C. dan Prasetyo, H., 2020. *Analisis dan Perancangan Sistem Informasi Menggunakan Flowchart dan Data Flow Diagram*. Jurnal Teknik Informatika.

Suli, S. dan Nirsal, N., 2023. *Implementasi Bootstrap dan SweetAlert2 dalam Pengembangan Sistem Informasi Berbasis Web*. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer.

Sidik, B., 2018. *Pemrograman Web dengan PHP 7*. Bandung: Informatika.

Supaartagorn, C., 2011. *PHP Framework for Database Management Based on MVC Pattern*. International Journal of Computer Science and Information Security.

Wicaksono, Y., 2019. *Membangun Aplikasi Web dengan Framework Laravel*. Yogyakarta: Lokomedia.

Ardhana, Y.K., 2020. *Pemrograman PHP: Teori dan Praktik Membuat Web Dinamis*. Jakarta: Andi Publisher.

Connolly, T.M. dan Begg, C.E., 2015. *Database Systems: A Practical Approach to Design, Implementation, and Management. Boston: Pearson Education.*

Fatta, H.A., 2019. *Analisis dan Perancangan Sistem Informasi untuk Keunggulan Bersaing Perusahaan dan Organisasi Modern*. Yogyakarta: Andi Offset

Dinas Kesehatan Kota Surabaya, 2024. *Profil Kesehatan Kota Surabaya Tahun 2024. Surabaya: Dinas Kesehatan Kota Surabaya*.

Rosa, A.S. dan Shalahuddin, M., 2018. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Bandung: Informatika.

Yuhefizar, 2019. *Cara Mudah Membangun Website Interaktif Menggunakan Content Management System*. Jakarta: Elex Media Komputindo.

Wardana, 2020. *Menguasai Framework Laravel untuk Pengembangan Aplikasi Web Modern*. Jakarta: Elex Media Komputindo.

Basuki, A.P., 2018. *Membangun Web Berbasis PHP dengan Framework CodeIgniter*. Yogyakarta: Lokomedia.

Purwanto, Y., 2019. *Pemrograman Web dengan PHP dan Framework Laravel*. Jakarta: Elex Media Komputindo.

Rosaly, C. dan Prasetyo, H., 2020. *Analisis dan Perancangan Sistem Informasi Menggunakan Flowchart dan Data Flow Diagram*. Jurnal Teknik Informatika.

Naista, W., 2020. *Sistem Informasi Pendaftaran Mahasiswa Baru Berbasis Web*. Jurnal Teknologi Informasi dan Ilmu Komputer.

Lampiran 1 : Surat Balasan

PEMERINTAH KOTA SURABAYA
DINAS KESEHATAN
JalanJemursari No. 197 Surabaya
Telepon(031) 8439473, 8439372
Laman surabaya.go.id, Pos-el: dinkes@surabaya.go.id

NOTA DINAS

Yth. : Sekretaris
Dari : Ka. Bidang Sumber Daya Kesehatan
Tembusan : 1. Kepala Tim Kerja Program Informasi Dan Hubungan Masyarakat
2. Dekan Fakultas Teknik Universitas 17 Agustus 1945 (untag) Surabaya
Tanggal : 29 Juli 2025
Nomor : 400.14.5.4 /8821/436.7.2/2025
Sifat : Biasa
Lampiran : -
Hal : Praktik Kerja Lapangan

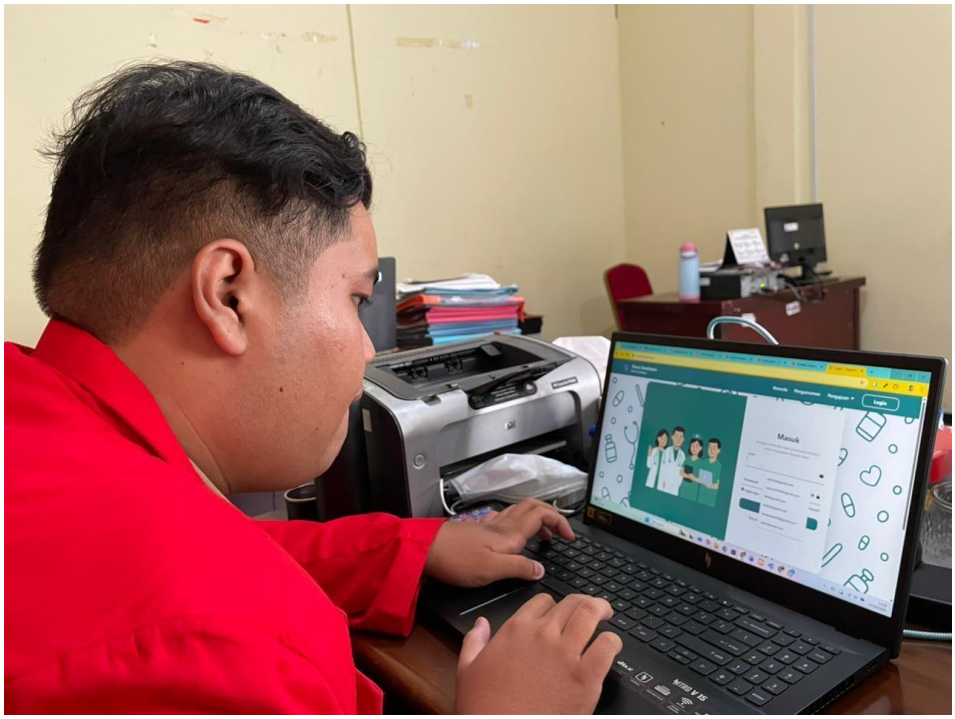
Memperhatikan Surat dari Dekan Fakultas Teknik Universitas 17 Agustus 1945 (UNTAG) Surabaya Nomor : 1079/K/FT/Akd/VII/2025 Tanggal : 11 Juli 2025 Hal Praktik Kerja Lapangan, dengan ini kami sampaikan bahwa Dinas Kesehatan Kota Surabaya dapat menerima Praktik Kerja Lapangan Mahasiswa a/n **Nalendra Putra W** Dan a/n **Muhammad Fadli** Dari Fakultas Teknik Universitas 17 Agustus 1945 (UNTAG) Surabaya pada tanggal : 1 Agustus 2025 - 31 Oktober 2025, bertempat di Tim Kerja Program
Informasi Dan Hubungan Masyarakat Dinas Kesehatan Kota Surabaya.

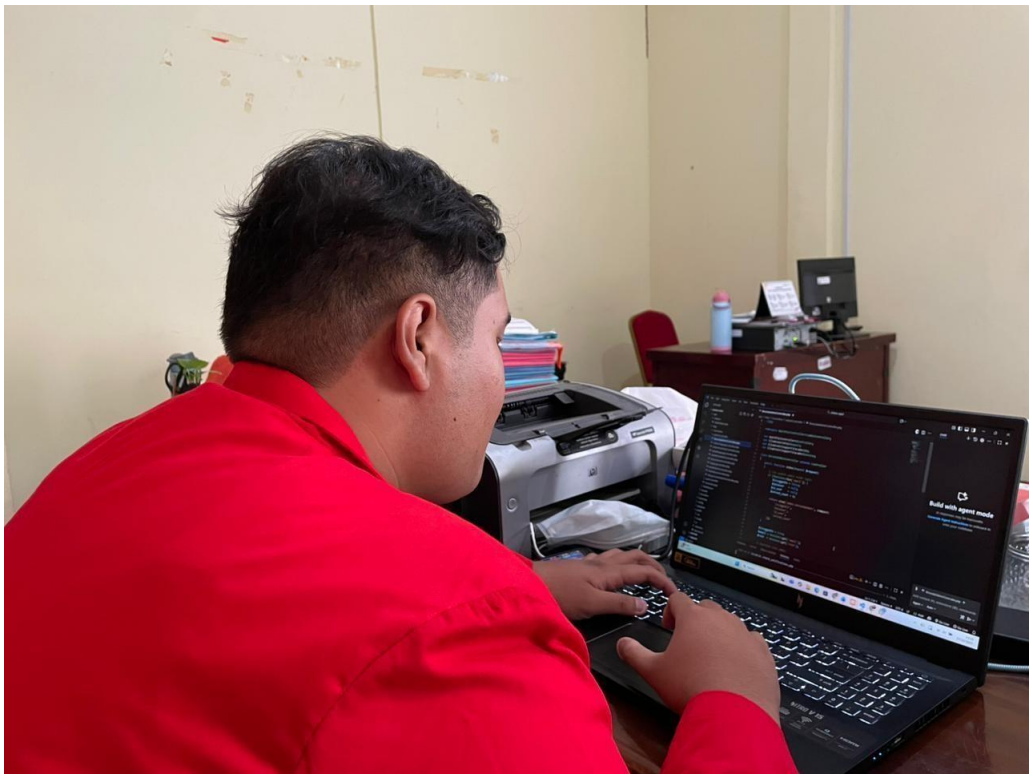
Demikian atas perhatiannya, disampaikan terima kasih.

KEPALA BIDANG,

dr. REYNER MEILAKSANA SUMBUNG
Pembina Tingkat I
NIP 197905192006041018

Lampiran 2 : Dokumentasi Kegiatan


Masuk
Login


Build with agent mode

Lampiran 3 : Form Kuisioner

KUESIONER UNTUK INSTITUSI PENGGUNA
MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
UNIVERSITAS 17 AGUSTUS 1945 SURABAYA

Program studi Teknik Informatika Universitas 17 Agustus 1945 Surabya mengadakan Survei mengenai Profile Mahasiswa Kerja Praktek. Tujuan dari Survei ini untuk mengevaluasi pengembangan kurikulum di Program studi Teknik Informatika Universitas 17 Agustus 1945 Surabya yang merupakan aktifitas penting untuk meningkatkan program studi. Hasil survei ini akan digunakan untuk bahan evaluasi pengembangan kurikulum di Program studi Teknik Informatika Universitas 17 Agustus 1945 Surabya. Kami mohon kesediaan Bapak/Ibu untuk menjawab survei ini. Terima kasih.

I. Biodata

Nama Mahasiswa	: Nalendra Putra Wicaksana
NIM	: 1462300034
Judul Kerja Praktek	: Rancang Bangun Backend Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya

II. Profile Umum

Nama Instansi	: Dinas Kesehatan Kota Surabaya
Alamat	: Jl. Raya Jemursari No.197, Sidosermo, Kec. Wonocolo, Surabaya, Jawa Timur 60239
No. Telepon	: 031 - 8439473
Homepage	: dinkes.surabaya.go.id.
Pembimbing Lapangan	: Dudy Heriyanto, S.Kom.
Jabatan	: Staff
Email	: doedi.heriyanto@gmail.com

III. Kompetensi

Berilah tanda ceklis yang paling sesuai untuk menggambarkan kompetensi Mahasiswa selama melaksanakan Kerja Praktek. Kompetensi pada saat mulai melaksanakan Kerja Praktek:

SB: Sangat Baik
B : Baik
C : Cukup
K : Kurang

Kategori	Penilaian			
	SB	B	C	K
1. Motivasi dalam menyelesaikan pekerjaan		v		
2. Kreativitas dalam menyelesaikan pekerjaan		✓		
3. Motivasi dalam menambah pengetahuan atau keahlian yang dimiliki		v		
4. Motivasi dalam menambah pengetahuan atau keahlian diluar bidang ilmu yang dimiliki		v		
5. Kemampuan dalam memecahkan permasalahan		v		
6. Kemampuan dalam menuangkan ide atau inovasi		v		
7. Kemampuan dalam berpikir logis		✓		
8. Kemampuan dalam menyelesaikan pekerjaan		v		
9. Kemampuan dalam melaporkan hasil pekerjaan	✓			
10. Kemampuan dalam menangani permasalahan		v		
11. Kemampuan dalam memenuhi segala aturan atau petunjuk kerja	✓			
12. Kemampuan dalam bekerja mandiri		v		
13. Kemampuan dalam mengerjakan pekerjaan yang sesuai bidang ilmu		v		
14. Kemampuan berkomunikasi dengan pimpinan		✓		
15. Kemampuan berkomunikasi dengan rekan kerja		v		
16. Etika dan moral di tempat kerja Praktek	✓			
17. Kemampuan dalam menyelesaikan pekerjaan rutin		✓		

Kategori	Penilaian			
	SB	B	C	K
18. Kemampuan dalam membantu rekan kerja	✓			
19. Kemampuan dalam menyelesaikan masalah tim	✓			
20. Kemampuan dalam berkerjasama dalam tim	✓			

Saran-saran terhadap Mahasiswa Kerja Praktek

Perbanyak referensi, sehingga akan ada ide dan inovasi lain

Saran-saran untuk perbaikan Program Studi Teknik Informatika Universitas 17 Agustus 1945 Surabaya

Ber inovasi dalam pembelajaran, karena dunia IT selalu berkembang dengan dinamis

Terimakasih atas partisipasi Saudara.

Surabaya, 27 Oktober 2025

.Pembimbing Lapangan

Dudy Heriyanto, S.Kom.

Lampiran 4 : Form Penilaian Kerja Praktek

FORMULIR PENILAIAN KERJA PRAKTEK
MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS 17 AGUSTUS 1945 SURABAYA

Nama Mahasiswa : Nalendra Putra Wicaksana
NIM : 1462300034
Judul Kerja Praktek : Rancang Bangun Backend Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya
Nama Instansi : Dinas Kesehatan Kota Surabaya
Alamat : Jl. Raya Jemursari No.197, Sidosermo, Kec. Wonocolo, Surabaya, Jawa Timur 60239
Waktu Pelaksanaan : 01 Agustus 2025 s.d 31 Oktober 2025

No	Penilaian	Bobot (B)	Nilai (N)	B x N
1	Kehadiran	20%	98	19,6
2	Kerjasama	20%	97	19,4
3	Komunikasi	10%	90	9
4	Sikap, Etika dan Tingkah Laku	20%	96	19,2
5	Prestasi Kerja	20%	95	19
6	Kreatifitas	10%	94	9,4
Jumlah				95,6

Surabaya, 27 Oktober 2025
.Pembimbing Lapangan

Dudy Heriyanto, S.Kom.

Lampiran 5 : Form Aktivitas Kerja Praktek

AKTIVITAS HARIAN KERJA PRAKTEK
MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS 17 AGUSTUS 1945 SURABAYA

Nama Mahasiswa : Nalendra Putra Wicaksana
NIM : 1462300034
Judul Kerja Praktek : Rancang Bangun Backend Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya

No	Tanggal	Keterangan	TTD
1	Jum'at, 01 Agustus 2025	Melakukan perencanaan terkait pembuatan Aplikasi Berbasis Web pada pendaftaran Magang dan Penelitian.	
2	Senin, 04 Agustus 2025	Melakukan Perancangan yang tediri dari pembuatan Use Case, Flowchart, Er Diagram, dan Proses Bisnis.	
3	Selasa, 05 Agustus 2025	Melakukan Pembuatan flowchart untuk menggambarkan proses pendaftaran magang dan penelitian.	
4	Kamis, 14 Agustus 2025	Melakukan pembuatan *Entity Relationship Diagram (ERD)* untuk mendefinisikan entitas, atribut, serta relasi antar tabel pada sistem basis data.	
5	Kamis, 20 Agustus 2025	Membuat *Use Case Diagram* untuk menggambarkan interaksi antara pengguna sistem seperti mahasiswa dan admin Dinas Kesehatan Kota Surabaya.	

6	Rabu, 27 Agustus 2025	Mengembangkan *Proses Bisnis Diagram* untuk menggambarkan alur kerja kegiatan pendaftaran magang dan penelitian.	
7	Selasa, 09 September 2025	Mulai melakukan implementasi *backend* menggunakan PHP Native pada web pendaftaran magang dan penelitian.	
8	Rabu 10 September 2025	Mengembangkan fitur pendaftaran magang dan penelitian menggunakan PHP Native.	
9	Senin 22 September 2025	Mulai tahap migrasi sistem ke framework Laravel untuk pengembangan lebih lanjut dan Mengonversi fitur pendaftaran dan autentikasi ke dalam Laravel.	

Surabaya, 27 Oktober 2025

.Pembimbing Lapangan

Dudy Heriyanto, S.Kom

Lampiran 6 : Lembar Bimbingan Kerja Praktek

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS 17 AGUSTUS 1945 SURABAYA

LEMBAR BIMBINGAN KERJA PRAKTEK

Semester Gasal / Genap Tahun 2025/2026 Periode : Gasal

Pas Photo 4 x 6	Nama	:	Nalendra Putra Wicaksana
	NBI	:	1462300034
	Alamat Rumah / Kost	:	Kota : Sidoarjo, Perum Griya Kartika blok u 12A Cemandi, Sedati, Sidoarjo, RT/RW : 21/05, Kelurahan : Cemandi, Kecamatan : Sedati, Kode Pos : 61253
	No Telp. / Hp	:	08993477653
	Pembimbing	:	Intan Dzikria (20460160701)
Mulai Bimbingan	Judul KP	:	Rancang Bangun Backend Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya
21 Agustus 2025			

PERSETUJUAN DOSEN PEMBIMBING	NILAI
Tanggal : 12 Desember 2025 Ttd. Pembimbing (Intan Dzikria, S.Kom., M.IM., Ph.D.)	

LEMBAR BIMBINGAN KERJA PRAKTEK

NO	HARI / TGL	URAIAN MATERI	TT.DOSEN
1	KAMIS/ 21-08-2025	PENGARAHAN PROYEK KEGIATAN KERJA PRAKTEK	
2	KAMIS/ 23-10-2025	BIMBINGAN DAN PENGARAHAN PENYUSUNAN PROPOSAL KERJA PARKTEK	
3	KAMIS/ 30-10-2025	PENGARAHAN DAN REVISI PROPOSAL KERJA PRAKTEK	
4	SENIN/ 03-11-2025	BIMBINGAN DAN PENGARAHAN LAPORAN KERJA PRAKTEK	
5	SENIN/ 01-12-2025	PENGARAHAN DAN REVISI LAPORAN KERJA PRAKTEK	

JUDUL KERJA PRAKTEK SETELAH DIREVISI
Rancang Bangun Backend Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya

LEMBAR PENGESAHAN JUDUL KERJA PRAKTEK	
Tanggal : 12 Desember 2025	
Ttd. Pembimbing	Ttd. Koordinator
Intan Dzikria, S.Kom., M.IM., Ph.D. NIP : 20460160701	Anton Breva Yunanda S.T., M.MT NPP. 20450020554

** Cetak dilembar buffalo kuning*

SYARAT MAJU PRESENTASI KERJA PRAKTEK :

1. Bimbingan Kerja Praktek minimal 3x

Lampiran 7 : Checklist Laporan Kerja Praktek

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS 17 AGUSTUS 1945 SURABAYA

CHECKLIST LAPORAN KERJA PRAKTEK

Semester Gasal / Genap Tahun 2025/2026 Periode : Gasals

Nama	:	Nalendra Putra Wicaksana
NBI	:	1462300034
Alamat Rumah / Kost	:	Kota : Sidoarjo, Perum Griya Kartika blok u 12A Cemandi, Sedati, Sidoarjo, RT/RW : 21/05, Kelurahan : Cemandi, Kecamatan : Sedati, Kode Pos : 61253
No Telp. / Hp	:	08993477653
Pembimbing	:	Intan Dzikria, S.Kom., M.IM., Ph.D. (20460160701)
Judul KP	:	Rancang Bangun Backend Sistem Pendaftaran Magang dan Penelitian Berbasis Web Dinas Kesehatan Kota Surabaya

Dosen Pembimbing wajib memberikan check (√) untuk tiap point yang telah dipenuhi.

Ketentuan umum yang harus dipenuhi

- ☑ Mahasiswa telah lulus mata kuliah minimal 72 sks
- ☑ Mahasiswa mempunyai IPK minimal 2.50
- ☑ Mahasiswa sudah mencantumkan mata kuliah Kerja Praktek dalam KRS
- ☑ Kerja Praktek sudah sesuai dengan bidang ilmu pada program studi Teknik Informatika
- ☑ Mahasiswa sudah melakukan pembayaran untuk mengikuti mata kuliah Kerja Praktek pada periode saat ini

Sistematika Penulisan Laporan

- ☑ Font yang digunakan adalah Times New Roman dengan ukuran 12
- ☑ Jarak baris pada laporan KP adalah 1.5 spasi
- ☑ Ukuran kertas yang digunakan adalah A4 dengan minimal 50 halaman
- ☑ Ukuran margin yang digunakan sudah sesuai aturan, yaitu right, top, bottom adalah 3 cm, dan left 4 cm
- ☑ Halaman Sampul sampai Daftar Isi diberi nomor halaman dengan huruf: i, ii, iii, …. dst dan diletakkan pada sudut kanan bawah
- ☑ Halaman Pendahuluan sampai Daftar Pustaka diberi nomor halaman dengan angka arab: 1, 2, 3, …dst yang diletakkan pada sudut kanan atas, kecuali untuk halaman yang mengandung judul bab diletakkan pada tengah halaman bawah

Surabaya, 12 Desember 2025

Mengetahui,	Dosen Pembimbing
Koordinator KP	
Anton Breva Yunanda S.T., M.MT	Intan Dzikiria, S.Kom., M.IM., Ph.D.
NPP. 20450020554	NIP : 20460160701