

LAPORAN PROYEK UAS PEMROGRAMAN WEB
ARVO – APLIKASI MANAJEMEN AKADEMIK MAHASISWA

Kelas:

2024D

Mata Kuliah:

Pemrograman Web



Oleh:

Anggun Amaylia Abdillah (24091397123)

Link youtube: https://youtu.be/8xXH48j8tE?si=SbDol2zaL_tsAYJ

Link GitHub: <https://github.com/anggun2024D/arvo-app.git>

Link Website: <https://arvo.infinityfreeapp.com/>

PROGRAM STUDI MANAJEMEN INFORMATIKA
FAKULTAS VOKASI
UNIVERSITAS NEGERI SURABAYA
2025

KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas limpahan rahmat dan karunia-Nya, sehingga saya dapat menyelesaikan laporan akhir Ujian Akhir Semester (UAS) Mata Kuliah Pemrograman Web dengan judul "ARVO: Aplikasi Manajemen Akademik Mahasiswa". Laporan ini merupakan hasil dari proses pengembangan proyek yang bertujuan untuk menciptakan solusi digital guna meningkatkan produktivitas mahasiswa dalam mengelola kegiatan akademik dan non-akademik mereka.

ARVO dirancang sebagai platform terintegrasi yang menyediakan fitur manajemen jadwal, pencatatan notulensi, penetapan target, serta visualisasi progres kegiatan. Proses pengembangan ini tidak hanya memenuhi persyaratan akademik, tetapi juga diharapkan memberikan kontribusi praktis bagi mahasiswa dalam menghadapi tantangan manajemen waktu di era digital.

Saya menyadari bahwa laporan ini masih memiliki keterbatasan. Oleh karena itu, saya terbuka terhadap kritik dan saran yang konstruktif untuk penyempurnaan di masa mendatang. Terima kasih kami sampaikan kepada Bapak Dosen Pengampu Mata Kuliah Pemrograman Web yaitu Bapak Ari Kurniawan yang telah memberikan bimbingan dan arahan selama proses ini. Ucapan terima kasih juga kami haturkan kepada teman-teman mahasiswa serta keluarga yang telah memberikan dukungan moral dan material.

Surabaya, 23 Desember 2025

Tim Pengembang ARVO

DAFTAR ISI

KATA PENGANTAR	1
DAFTAR ISI.....	1
BAB I.....	3
PENDAHULUAN	3
1.1 Latar Belakang	3
1.2 Rumusan Masalah	4
1.3 Tujuan Pengembangan	4
1.4 Ruang Lingkup.....	5
BAB II.....	7
ANALISIS SISTEM	7
2.1 Pendahuluan	7
2.2 Daftar Fitur.....	7
2.3 Lingkungan Sistem.....	9
2.4 Daftar Pengguna.....	10
2.5 Kebutuhan Fungsional.....	10
2.6 Kebutuhan Non-Fungsional	13
2.7 Permasalahan dan Analisis.....	15
BAB III	24
PERANCANGAN SISTEM	24
3.1 Arsitektur Sistem.....	24
3.2 Desain Database	34
3.3 Perancangan Tampilan	37
BAB IV	46
IMPLEMENTASI WEBSITE	46

4.1 Struktur Folder	46
4.2 Kode Program yang Penting	48
BAB V	84
HASIL & PEMBAHASAN	84
5.1 Hasil Pengujian Sistem.....	84
5.2 Pembahasan Fungsi Sistem	85
BAB VI.....	87
PENUTUP.....	87
6.1 Kesimpulan.....	87
6.2 Saran Pengembangan	88

BAB I

PENDAHULUAN

1.1 Latar Belakang

Di era pendidikan tinggi yang semakin kompetitif, mahasiswa sering kali dihadapkan pada berbagai tuntutan aktivitas yang kompleks, meliputi perkuliahan, partisipasi dalam organisasi kemahasiswaan, penyelesaian tugas akademik, serta pengembangan diri melalui kegiatan pribadi. Manajemen waktu yang efektif menjadi faktor krusial dalam menentukan keberhasilan akademik dan kesejahteraan pribadi mahasiswa. Namun, berdasarkan penelitian dari Steel (2007), sekitar 80-95% mahasiswa mengalami prokrastinasi akademik yang disebabkan oleh kurangnya kemampuan manajemen waktu, yang pada akhirnya berdampak pada penurunan prestasi belajar dan peningkatan stres.

Permasalahan ini semakin diperburuk oleh metode pengelolaan aktivitas yang masih manual, seperti penggunaan catatan kertas atau aplikasi umum (misalnya, Google Keep atau Calendar) yang tidak dirancang secara khusus untuk kebutuhan mahasiswa. Akibatnya, mahasiswa sering mengalami jadwal yang bertabrakan, kelupaan terhadap deadline tugas, kesulitan dalam melacak progres target jangka panjang, serta minimnya dokumentasi kegiatan yang terstruktur. Penelitian dari Lestari (2020) menunjukkan bahwa mahasiswa dengan perilaku manajemen waktu yang buruk cenderung memiliki efektivitas belajar yang rendah, dengan pengaruh signifikan terhadap IPK. Selain itu, survei dari Universitas Negeri Surabaya (2024) mengindikasikan bahwa 70% mahasiswa mengalami kesulitan dalam menyeimbangkan aktivitas akademik dan organisasi karena kurangnya tools integratif.

Perkembangan teknologi informasi, khususnya pemrograman web, membuka peluang untuk menciptakan solusi digital yang lebih efisien. Aplikasi berbasis web menawarkan keunggulan dalam hal aksesibilitas multi-perangkat, penyimpanan data terpusat, dan kemampuan analitik real-time. Berdasarkan latar belakang tersebut, dikembangkan ARVO (Activity & Goals Manager), sebuah aplikasi web yang dirancang khusus untuk mahasiswa. ARVO mengintegrasikan fitur-fitur utama seperti manajemen jadwal dengan prioritas, pencatatan notulensi terstruktur, penetapan target dengan visualisasi progres, serta dashboard analitik untuk memberikan insight produktivitas. Pendekatan user-centric design diterapkan untuk memastikan kemudahan penggunaan, sementara teknologi seperti PHP dan MySQL digunakan untuk keandalan sistem.

Pengembangan ARVO tidak hanya memenuhi kebutuhan praktis mahasiswa, tetapi juga sejalan dengan tujuan pendidikan tinggi untuk membekali mahasiswa dengan keterampilan digital. Aplikasi ini diharapkan dapat mengurangi prokrastinasi, meningkatkan efisiensi waktu, dan mendukung pencapaian tujuan akademik secara holistik.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka rumusan masalah dalam pengembangan aplikasi ini adalah sebagai berikut:

1. Bagaimana merancang sistem manajemen jadwal yang fleksibel untuk mengakomodasi berbagai jenis aktivitas mahasiswa (kuliah, organisasi, tugas, dan pribadi) dengan mekanisme prioritas dan kategorisasi yang efektif?
2. Bagaimana mengimplementasikan fitur pencatatan notulensi yang mendukung dokumentasi terstruktur dan pencarian cepat untuk informasi dari perkuliahan atau rapat organisasi?
3. Bagaimana mengembangkan sistem penetapan dan pelacakan target yang menyediakan visualisasi progres secara real-time untuk memotivasi mahasiswa dalam mencapai tujuan akademik dan karir?
4. Bagaimana merancang dashboard analitik yang memberikan insight mendalam tentang pola aktivitas, distribusi waktu, dan indikator produktivitas mahasiswa?
5. Bagaimana memastikan keamanan data pribadi melalui implementasi sistem autentikasi, otorisasi, dan manajemen profil yang robust?
6. Bagaimana menciptakan antarmuka pengguna yang responsif, intuitif, dan dapat diakses dari berbagai perangkat, termasuk dukungan untuk mode gelap guna meningkatkan kenyamanan penggunaan?

Rumusan masalah ini difokuskan pada aspek fungsional, non-fungsional, dan usability untuk memastikan solusi yang komprehensif.

1.3 Tujuan Pengembangan

Tujuan pengembangan aplikasi ARVO dibagi menjadi tujuan umum dan tujuan khusus, sebagai berikut:

1. Tujuan Umum:
Mengembangkan aplikasi web terintegrasi yang mendukung mahasiswa dalam mengelola aktivitas akademik dan non-akademik secara efektif, sehingga meningkatkan produktivitas dan mengurangi stres akibat buruknya manajemen waktu.

2. Tujuan Khusus:

- Membangun modul manajemen jadwal dengan fitur kategorisasi (kuliah, organisasi, tugas, pribadi), prioritas (tinggi, sedang, rendah), dan reminder sederhana untuk mengoptimalkan pengelolaan aktivitas harian.
- Mengimplementasikan fitur notulensi digital yang memungkinkan pencatatan, pengeditan, dan pencarian informasi penting dari kuliah atau rapat dengan kategorisasi yang jelas.
- Menyediakan sistem penetapan target dengan tipe (semester, akademik, karir, personal), deadline, dan tracker progres (0-100%) untuk memotivasi pencapaian tujuan jangka panjang.
- Mengembangkan dashboard analitik yang menampilkan statistik kunci, grafik distribusi kegiatan, dan ringkasan progres untuk memberikan feedback visual kepada pengguna.
- Memastikan keamanan sistem melalui autentikasi berbasis session, hashing password, dan manajemen profil yang aman.
- Merancang antarmuka responsif dengan dukungan dark mode, navigasi intuitif, dan kompatibilitas multi-perangkat untuk meningkatkan aksesibilitas.
- Membangun basis data relasional yang efisien untuk menyimpan dan mengelola data pengguna, jadwal, notulensi, dan target.
- Mengintegrasikan kalender visual dengan navigasi bulanan dan indikator event untuk memudahkan perencanaan.
- Menyediakan fitur ekspor data dan penghapusan akun untuk mendukung portabilitas dan privasi pengguna.

1.4 Ruang Lingkup

Ruang lingkup pengembangan aplikasi ARVO mencakup batasan sistem, fungsional, dan non-fungsional untuk memfokuskan proyek pada tujuan utama. Berikut adalah penjelasan rinci:

A. Batasan Sistem

1. Pengguna:

- Ditargetkan untuk mahasiswa perguruan tinggi sebagai pengguna individu.
- Setiap akun bersifat pribadi tanpa fitur kolaborasi multi-user.

2. Fitur Utama:

- Manajemen jadwal dengan operasi CRUD (Create, Read, Update, Delete).
- Pencatatan notulensi dengan kategorisasi.

- Penetapan target dengan tracker progres.
 - Dashboard analitik dan kalender visual.
 - Manajemen profil dan autentikasi.
3. Platform:
- Berbasis web, diakses melalui browser tanpa instalasi.
 - Responsif untuk desktop, tablet, dan mobile (minimal resolusi 360x640 px).
4. Teknologi:
- Frontend: HTML5, CSS3, JavaScript (ES6+).
 - Backend: PHP 7.4+ dengan MySQL.
 - Server: Apache via XAMPP untuk pengembangan lokal.

B. Batasan Fungsional

1. Tidak menyediakan notifikasi real-time (push/email/SMS).
2. Tidak ada integrasi dengan layanan eksternal seperti Google Calendar atau cloud storage.
3. Tidak mendukung attachment file pada notulensi atau target.
4. Tidak ada fitur berbagi data antar pengguna.
5. Ekspor data terbatas pada format JSON tanpa otomatisasi backup.

C. Batasan Non-Fungsional

1. Kompatibilitas: Dioptimalkan untuk browser modern (Chrome 90+, Firefox 88+, dll.); tidak mendukung browser lama.
2. Performa: Dirancang untuk hingga 1000 entri data per pengguna; tidak untuk skala enterprise.
3. Keamanan: Menggunakan hashing dasar; tidak termasuk enkripsi lanjutan seperti SSL untuk produksi.
4. Lingkungan: Pengembangan lokal; deployment produksi di luar ruang lingkup.

Batasan ini ditetapkan untuk menjaga fokus proyek UAS sambil memastikan kelayakan implementasi dalam waktu terbatas.

1.5 Pembagian Tugas

No.	Nama	NIM	Peran/Tugas
1	Anggun Amaylia A.	24091397123	Frontend Developer, Backend Developer, Dokumentasi & Laporan

BAB II

ANALISIS SISTEM

2.1 Pendahuluan

Tahap analisis sistem merupakan langkah awal yang sangat penting dalam proses pengembangan aplikasi ARVO. Pada tahap ini dilakukan identifikasi kebutuhan pengguna, permasalahan yang dihadapi, serta tujuan sistem yang akan dibangun. Analisis sistem bertujuan untuk memastikan bahwa aplikasi yang dikembangkan benar-benar sesuai dengan kebutuhan pengguna dan mampu memberikan solusi yang efektif terhadap permasalahan yang ada.

Aplikasi ARVO dikembangkan sebagai sistem manajemen aktivitas mahasiswa yang terintegrasi, mencakup pengelolaan jadwal, notulensi, target, serta visualisasi aktivitas dalam bentuk kalender dan dashboard statistik. Sebelum sistem ini dibuat, mahasiswa umumnya masih mengandalkan pencatatan manual atau menggunakan aplikasi terpisah yang tidak saling terintegrasi, sehingga menyebabkan data aktivitas menjadi tidak terstruktur dan sulit dipantau secara menyeluruh.

Metode pengembangan sistem yang digunakan pada aplikasi ARVO adalah metode incremental. Metode ini dipilih karena memungkinkan pengembangan sistem dilakukan secara bertahap berdasarkan modul atau fitur tertentu, seperti autentikasi pengguna, manajemen jadwal, notulensi, target, hingga analitik dashboard. Dengan metode incremental, setiap fitur dapat diuji dan disempurnakan sebelum melanjutkan ke fitur berikutnya, sehingga risiko kesalahan dapat diminimalkan.

2.2 Daftar Fitur

Aplikasi ARVO memiliki sejumlah fitur utama yang dirancang untuk mendukung kebutuhan manajemen aktivitas mahasiswa secara terstruktur dan terintegrasi. Adapun fitur-fitur yang tersedia dalam aplikasi ARVO adalah sebagai berikut:

A. Fitur Autentikasi & Keamanan

No	Fitur	Deskripsi
1	Registrasi Pengguna	Pendaftaran akun dengan validasi email unik dan password minimal 8 karakter.
2	Login Sistem	Autentikasi berbasis username/password dengan session persistence.
3	Logout	Penghapusan session dan redirect ke halaman login.
4	Password	Ubah password dengan verifikasi password lama dan

	Management	konfirmasi baru.
5	Profile Management	Update data pribadi seperti nama, email, institusi, jurusan, dan bio.

B. Fitur Manajemen Jadwal

No	Fitur	Deskripsi
6	Tambah Jadwal	Input judul, tanggal, waktu, jenis (4 kategori), prioritas (3 level), deskripsi.
7	Edit Jadwal	Modifikasi data jadwal existing via modal form.
8	Hapus Jadwal	Penghapusan dengan konfirmasi untuk menghindari kesalahan.
9	Tandai Selesai	Toggle status completed untuk tracking penyelesaian.
10	Filter Jadwal	Filter dinamis berdasarkan kategori, prioritas, dan status.

C. Fitur Notulensi

No	Fitur	Deskripsi
11	Buat Notulensi	Input judul, kategori (4 jenis), dan konten teks panjang.
12	Edit Notulensi	Modifikasi konten via editor sederhana.
13	Hapus Notulensi	Penghapusan dengan konfirmasi.
14	Pencarian	Search real-time berdasarkan judul atau konten menggunakan JS.

D. Fitur Target

No	Fitur	Deskripsi
15	Set Target	Input tipe (4 jenis), judul, deskripsi, deadline, progres awal.
16	Update Progress	Slider untuk mengubah progres (0-100%) dengan update real-time.
17	Update Target	Modifikasi data target via modal.
18	Hapus Target	Penghapusan dengan konfirmasi.

E. Fitur Dashboard & Analytics

No	Fitur	Deskripsi
19	Quick Stats	6 metrik: Total jadwal, notulensi, tugas selesai, hari aktif,

		prioritas tinggi, progres target.
20	Kegiatan Hari Ini	List jadwal hari ini dengan sorting waktu.
21	Progress Overview	Ringkasan persentase keseluruhan dengan progress bar.
22	Analytics Chart	Grafik bar untuk distribusi kategori menggunakan CSS/JS.
23	Motivasi Harian	Quote acak dari array predefined.
24	Mini Calendar	Kalender mingguan dengan indikator event.

F. Fitur Kalender

No	Fitur	Deskripsi
25	Kalender Bulanan	Grid tampilan bulan dengan navigasi.
26	Event Indicator	Dot warna berdasarkan kategori pada tanggal terkait.
27	Detail Event	Modal popup untuk detail jadwal saat klik tanggal.

G. Fitur Tambahan

No	Fitur	Deskripsi
28	Dark Mode	Toggle tema menggunakan CSS variables.
29	Responsive Design	Media queries untuk adaptasi layar.
30	Export Data	Download JSON seluruh data pengguna.
31	Delete Account	Hapus akun dengan konfirmasi multi-step.

2.3 Lingkungan Sistem

Lingkungan sistem menjelaskan perangkat lunak dan perangkat keras yang digunakan dalam proses pengembangan dan pengujian aplikasi ARVO.

Lingkungan Perangkat Lunak (Software):

- XAMPP sebagai web server lokal yang menyediakan Apache dan MySQL.
- Visual Studio Code (VSCode) sebagai text editor untuk penulisan kode program.
- Google Chrome sebagai browser untuk pengujian aplikasi.
- GitHub sebagai media penyimpanan dan pengelolaan repositori kode sumber.

Lingkungan Perangkat Keras (Hardware):

- Laptop atau komputer dengan spesifikasi minimum:
 - RAM minimal 4GB
 - Prosesor setara Intel Core i3
 - Penyimpanan minimal 10 GB ruang kosong

Teknologi yang Digunakan:

- HTML untuk struktur halaman web.
- CSS untuk desain dan tampilan antarmuka.
- JavaScript untuk interaksi dan logika frontend.
- PHP untuk pemrosesan backend.
- MySQL sebagai sistem manajemen basis data.

2.4 Daftar Pengguna

ARVO dirancang dengan model single-role untuk kesederhanaan dan privasi.

A. Mahasiswa (User Utama)

Karakteristik:

- Pengguna individu dengan akun pribadi.
- Akses penuh ke data sendiri tanpa interaksi multi-user.

Hak Akses:

Modul	Create	Read	Update	Delete
Akun	✓	✓	✓	✓
Jadwal	✓	✓	✓	✓
Notulensi	✓	✓	✓	✓
Target	✓	✓	✓	✓
Dashboard	-	✓	-	-
Kalender	-	✓	-	-
Profil	-	✓	✓	-
Export Data	-	✓	-	-

Batasan:

- Data isolasi per akun untuk privasi.
- Tidak ada role admin; semua pengguna setara.

Justifikasi:

Desain single-user mengurangi kompleksitas, sesuai dengan use case personal productivity, dan mencegah risiko kebocoran data.

2.5 Kebutuhan Fungsional

Kebutuhan fungsional merupakan fungsi-fungsi yang harus disediakan oleh sistem agar dapat berjalan sesuai tujuan. Kebutuhan fungsional aplikasi ARVO meliputi:

A. Modul Autentikasi

ID	Kebutuhan Fungsional	Prioritas
F-01	Menerima registrasi dengan validasi email dan password.	High
F-02	Autentikasi login dengan session.	High
F-03	Logout dan session destruction.	High
F-04	Update password dengan verifikasi.	High
F-05	Proteksi halaman utama dari akses tidak autentik.	High

B. Modul Manajemen Jadwal

ID	Kebutuhan Fungsional	Prioritas
F-07	Sistem harus dapat menambahkan jadwal baru dengan input: judul, tanggal, waktu, jenis, prioritas, deskripsi	High
F-08	Sistem harus menyediakan 4 kategori jadwal: Kuliah, Organisasi, Tugas, Pribadi	High
F-09	Sistem harus menyediakan 3 level prioritas: Tinggi, Sedang, Rendah	High
F-10	Sistem harus dapat menampilkan daftar jadwal dengan filter berdasarkan kategori dan prioritas	High
F-11	Sistem harus dapat mengubah informasi jadwal yang sudah ada	High
F-12	Sistem harus dapat menghapus jadwal	High
F-13	Sistem harus dapat menandai jadwal sebagai selesai/belum selesai	Medium
F-14	Sistem harus menampilkan jadwal yang sudah selesai dengan tampilan berbeda	Medium

C. Modul Notulensi

ID	Kebutuhan Fungsional	Prioritas
F-15	Sistem harus dapat menambahkan notulensi dengan input: judul, kategori, isi konten	High
F-16	Sistem harus menyediakan 4 kategori notulensi: Kuliah, Organisasi, Pribadi, Lainnya	High
F-17	Sistem harus dapat menampilkan daftar semua notulensi	High

	dengan timestamp	
F-18	Sistem harus menyediakan fitur pencarian notulensi berdasarkan judul dan isi	Medium
F-19	Sistem harus dapat mengubah isi notulensi yang sudah ada	High
F-20	Sistem harus dapat menghapus notulensi	High

D. Modul Target

ID	Kebutuhan Fungsional	Prioritas
F-21	Sistem harus dapat menambahkan target dengan input: tipe, judul, deskripsi, deadline, progress	High
F-22	Sistem harus menyediakan 4 tipe target: Semester, Akademik, Karir, Personal	High
F-23	Sistem harus dapat menampilkan visualisasi progress target dalam bentuk progress bar	High
F-24	Sistem harus dapat mengubah progress target (0-100%)	High
F-25	Sistem harus dapat mengubah informasi target yang sudah ada	High
F-26	Sistem harus dapat menghapus target	High

E. Modul Dashboard

ID	Kebutuhan Fungsional	Prioritas
F-27	Sistem harus menampilkan 6 statistik: Total Jadwal, Total Notulensi, Tugas Selesai, Hari Aktif, Prioritas Tinggi, Progress Target	High
F-28	Sistem harus menampilkan daftar kegiatan hari ini yang diurutkan berdasarkan waktu	High
F-29	Sistem harus menampilkan ringkasan progress keseluruhan	Medium
F-30	Sistem harus menampilkan analytics distribusi kegiatan dalam bentuk chart/grafik	Medium
F-31	Sistem harus menampilkan quote motivasi yang berubah setiap hari	Low
F-32	Sistem harus menampilkan mini calendar mingguan dengan event indicator	Medium

F. Modul Kalender

ID	Kebutuhan Fungsional	Prioritas
F-33	Sistem harus menampilkan kalender bulanan dengan grid tanggal	High
F-34	Sistem harus menampilkan indicator (dot) pada tanggal yang memiliki jadwal	High
F-35	Sistem harus dapat navigasi ke bulan sebelumnya dan berikutnya	High
F-36	Sistem harus menampilkan detail jadwal saat tanggal diklik	High
F-37	Sistem harus memberikan color coding berbeda untuk setiap kategori kegiatan	Medium

G. Modul Profil

ID	Kebutuhan Fungsional	Prioritas
F-38	Sistem harus dapat menampilkan informasi profil pengguna	High
F-39	Sistem harus dapat mengubah informasi profil: nama, email, institusi, jurusan, bio	High
F-40	Sistem harus dapat mengubah password dengan validasi password lama	High
F-41	Sistem harus dapat mengekspor semua data pengguna dalam format JSON	Medium
F-42	Sistem harus dapat menghapus akun beserta semua data terkait	Medium

2.6 Kebutuhan Non-Fungsional

Kebutuhan non-fungsional menjelaskan kualitas sistem agar dapat digunakan secara optimal. Adapun kebutuhan non-fungsional pada aplikasi ARVO adalah sebagai berikut:

ID	Kategori	Kebutuhan	Kriteria
NF-01	Usability (Kegunaan)	Interface harus intuitif dan mudah dipelajari	Pengguna baru dapat menggunakan fitur dasar dalam 5 menit
NF-02		Navigasi antar halaman harus jelas dan konsisten	Maksimal 3 klik untuk mengakses fitur apapun

NF-03		Feedback visual harus jelas untuk setiap aksi	Notifikasi muncul dalam 0.5 detik
NF-04		Desain harus responsif untuk berbagai ukuran layar	Support minimal 360px (mobile) hingga 1920px (desktop)
NF-05		Font dan warna harus readable dan accessible	Contrast ratio minimal 4.5:1 (WCAG AA)
NF-06	Performance (Kinerja)	Halaman harus load dengan cepat	Waktu loading maksimal 3 detik
NF-07		Operasi CRUD harus responsif	Response time maksimal 1 detik
NF-08		Sistem harus dapat menangani data dalam jumlah besar	Support minimal 1000 jadwal per user tanpa lag
NF-09		Animasi dan transisi harus smooth	Frame rate minimal 30 fps
NF-10	Security (Keamanan)	Password harus di-hash sebelum disimpan	Menggunakan algoritma bcrypt atau sha256
NF-11		Session harus expire setelah inaktif	Timeout 30 menit
NF-12		Input harus divalidasi dan disanitasi	Prevent SQL injection dan XSS
NF-13		Akses halaman harus terproteksi	Redirect ke login jika belum autentikasi
NF-14	Reliability (Keandalan)	Sistem harus stabil dan tidak crash	Uptime minimal 95%
NF-15		Error harus di-handle dengan graceful	Menampilkan pesan error yang informatif
NF-16		Data harus konsisten	Tidak ada data loss saat operasi CRUD
NF-17	Maintainability (Pemeliharaan)	Kode harus modular dan terstruktur	Setiap modul memiliki fungsi spesifik
NF-18		Database harus normalized	Minimal 3NF
NF-19		Dokumentasi kode harus	Comment untuk fungsi

		lengkap	kompleks
NF-20	Compatibility (Kompatibilitas)	Cross-browser compatibility	Support Chrome, Firefox, Safari, Edge versi terbaru
NF-21		Mobile compatibility	Responsive di iOS dan Android browser
NF-22		Backward compatibility	Support browser 2 tahun ke belakang

Kebutuhan non-fungsional ini memastikan bahwa ARVO tidak hanya berfungsi dengan baik, tetapi juga memberikan pengalaman pengguna (*user experience*) yang optimal.

2.7 Permasalahan dan Analisis

2.7.1. Identifikasi Permasalahan

Berdasarkan analisis kebutuhan dan studi literatur, berikut permasalahan yang ditemukan:

1. Fragmentasi Tools:
 - Mahasiswa menggunakan berbagai aplikasi terpisah (Google Calendar untuk jadwal, Notes untuk catatan, spreadsheet untuk tracking)
 - Tidak ada integrasi antar tools
 - Data tersebar dan sulit dikelola
2. Kompleksitas Manajemen:
 - Kesulitan memprioritaskan aktivitas
 - Lupa deadline atau jadwal penting
 - Tidak ada visualisasi progress yang jelas
3. Kurangnya Insight:
 - Tidak tahu distribusi waktu untuk setiap aktivitas
 - Sulit mengidentifikasi pola produktivitas
 - Tidak ada feedback tentang pencapaian target
4. Aksesibilitas:
 - Tools yang ada sering berbayar atau memiliki paywall
 - Tidak semua tools mobile-friendly
 - Memerlukan instalasi aplikasi terpisah

2.7.2. Solusi yang Ditawarkan

ARVO dirancang untuk mengatasi permasalahan di atas dengan pendekatan:

1. Unified Platform:

- Satu aplikasi untuk semua kebutuhan (jadwal, notulensi, target)
- Data terintegrasi dan saling terhubung
- Akses terpusat melalui dashboard

2. Smart Organization:

- Kategorisasi otomatis dengan 4 kategori utama
- Sistem prioritas 3 level
- Filter dan search yang powerful

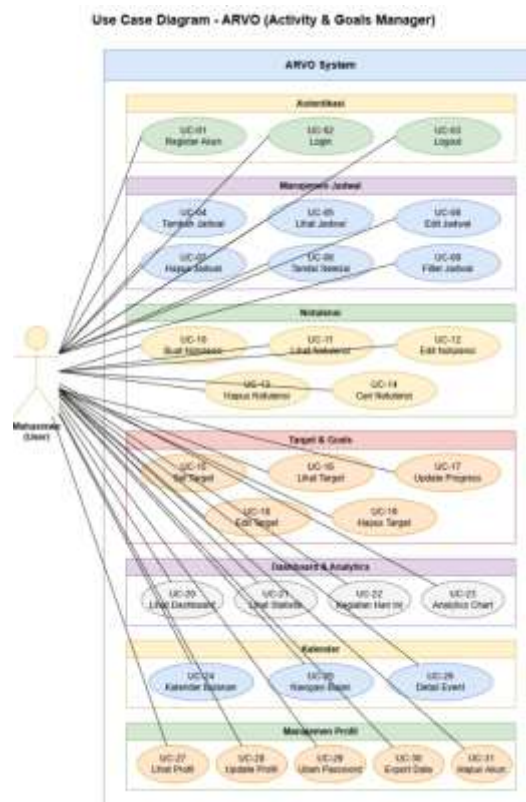
3. Visual Analytics:

- Dashboard dengan 6 quick stats
- Grafik distribusi aktivitas
- Progress bar untuk setiap target
- Kalender visual dengan color coding

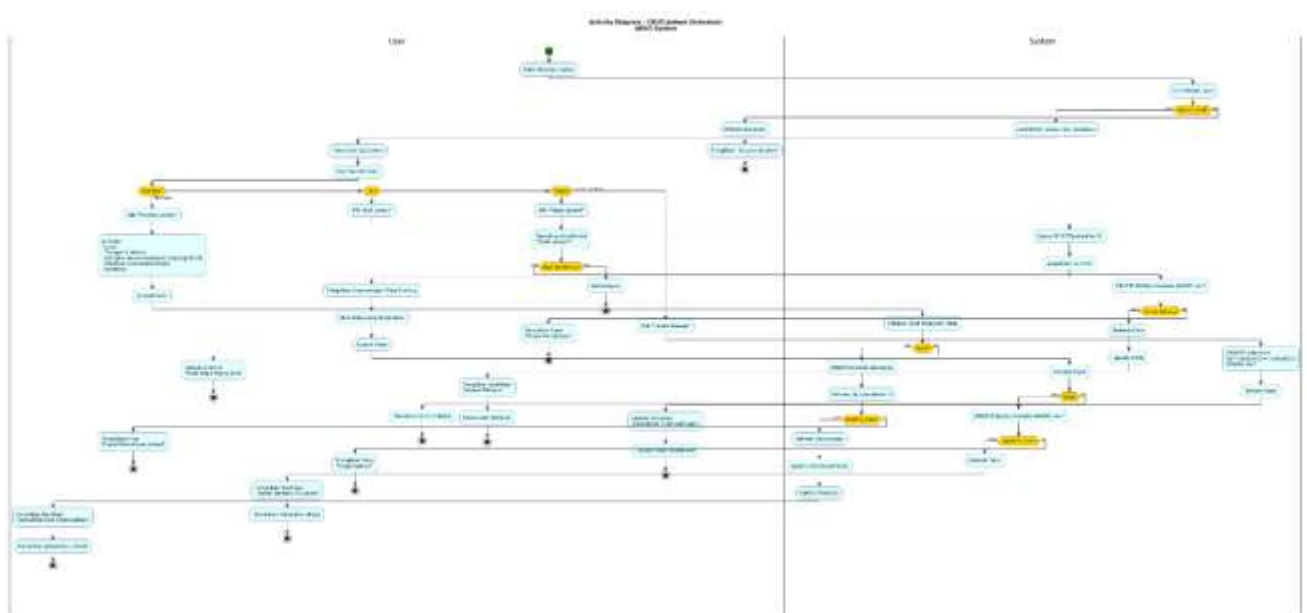
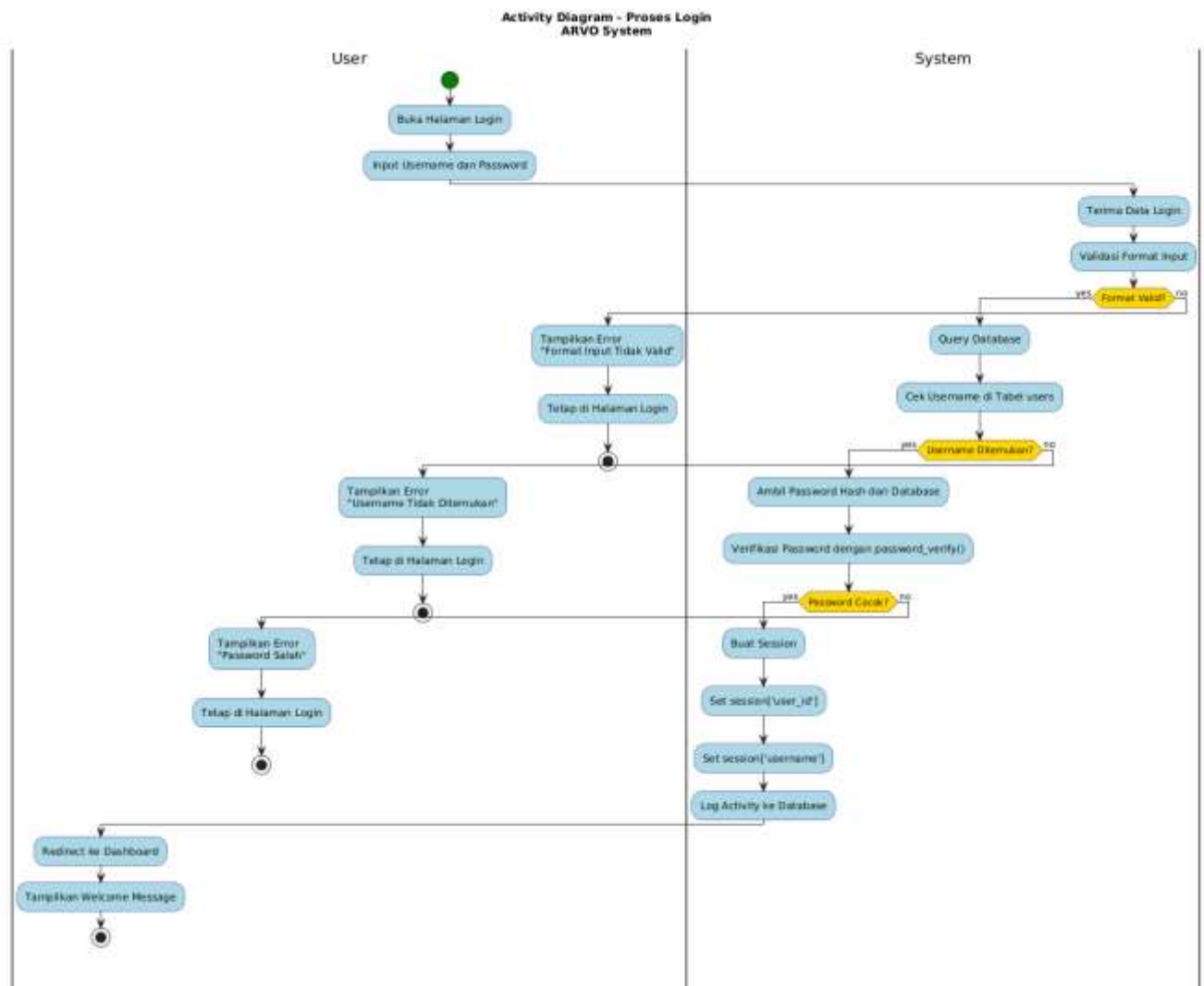
4. Aksesibilitas:

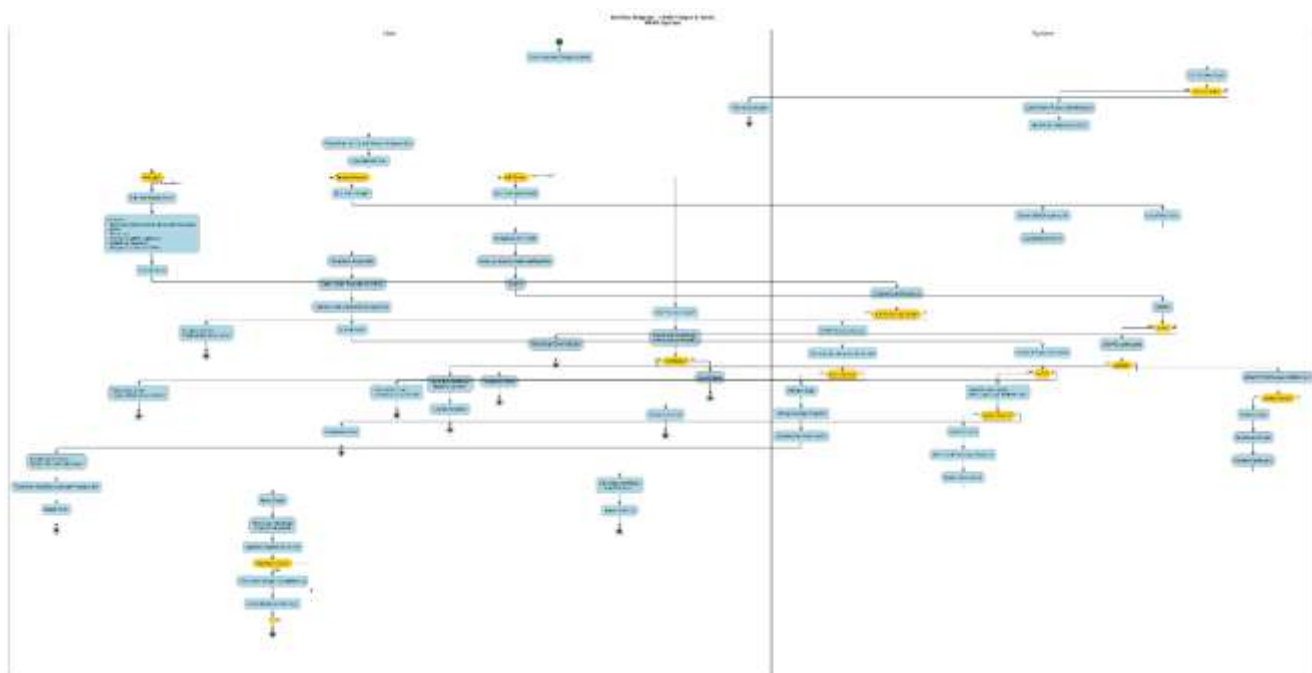
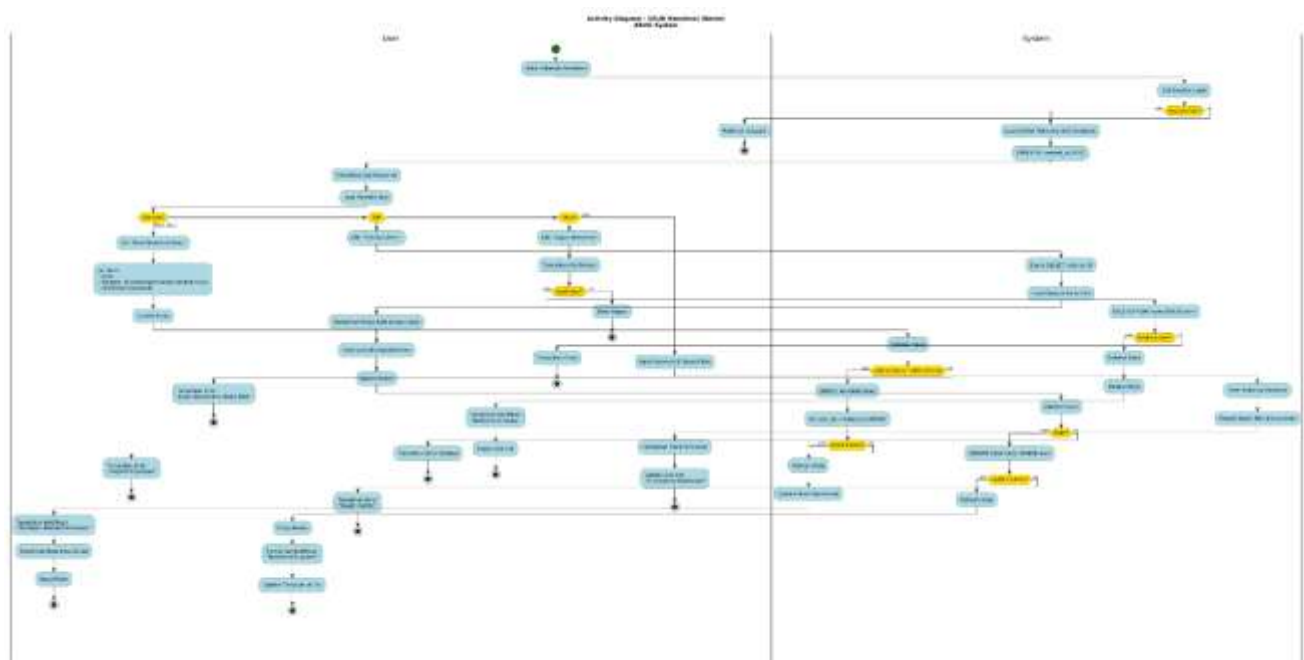
- 100% gratis tanpa paywall
- Web-based, tidak perlu instalasi
- Responsive design untuk semua device terpisah

2.7.3. Use Case Diagram

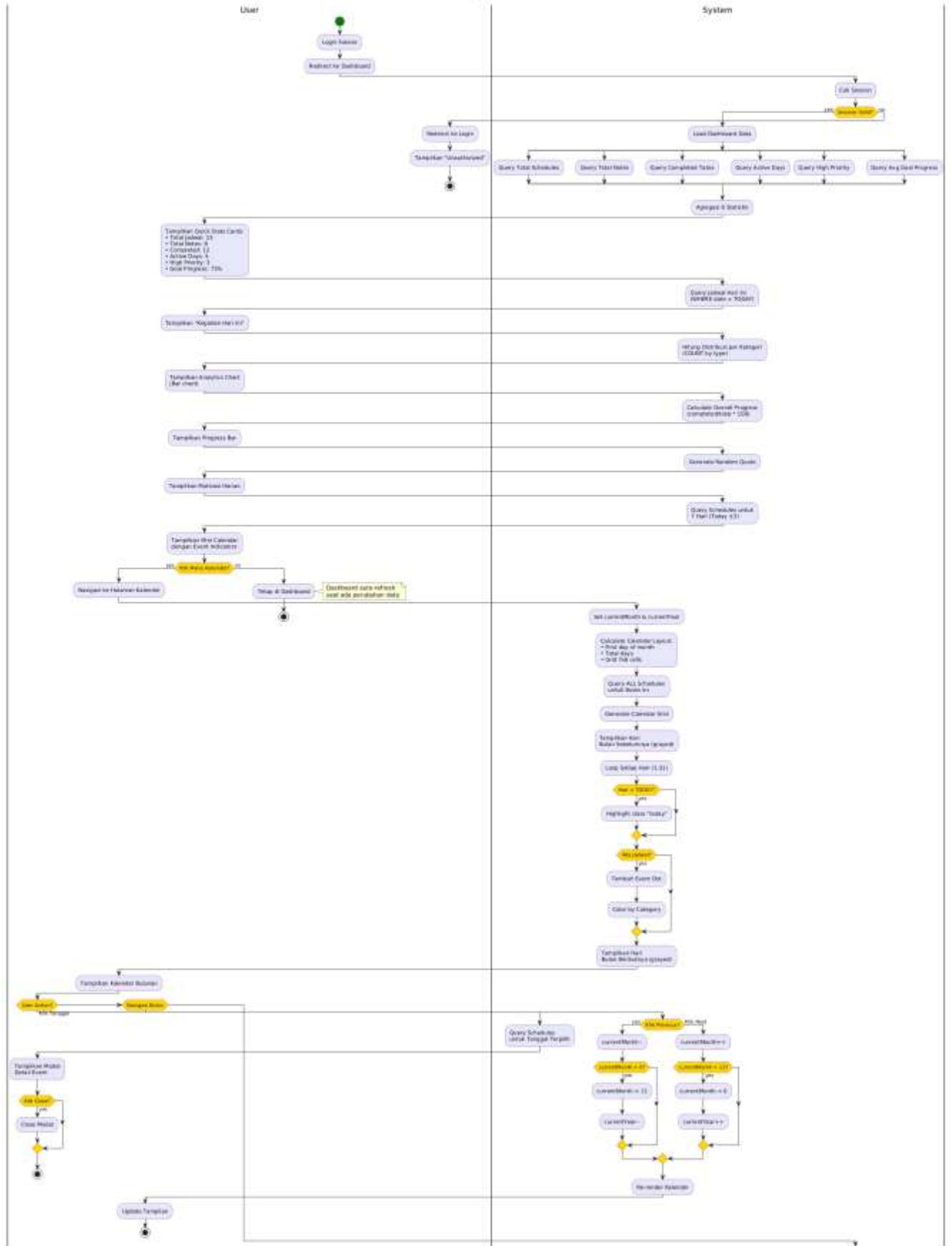


2.7.4. Activity Diagram

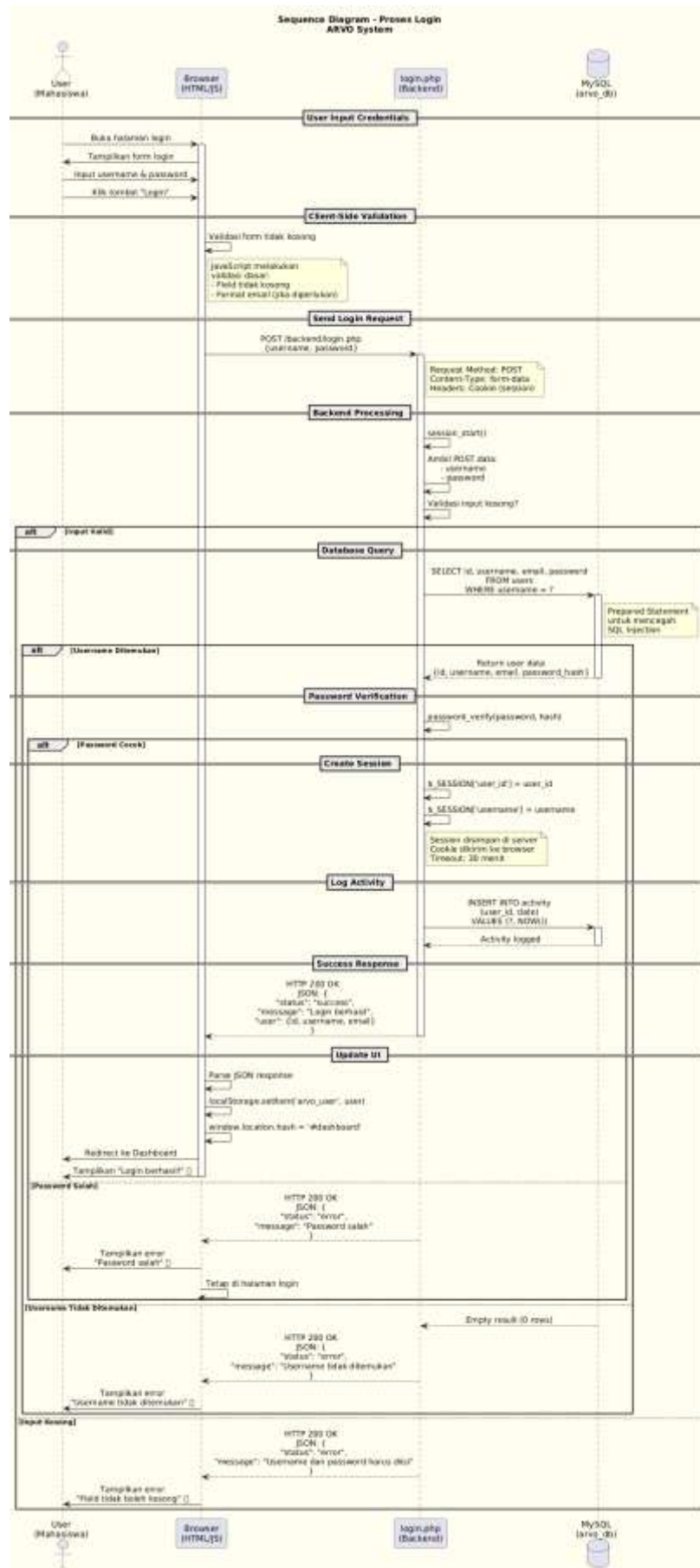




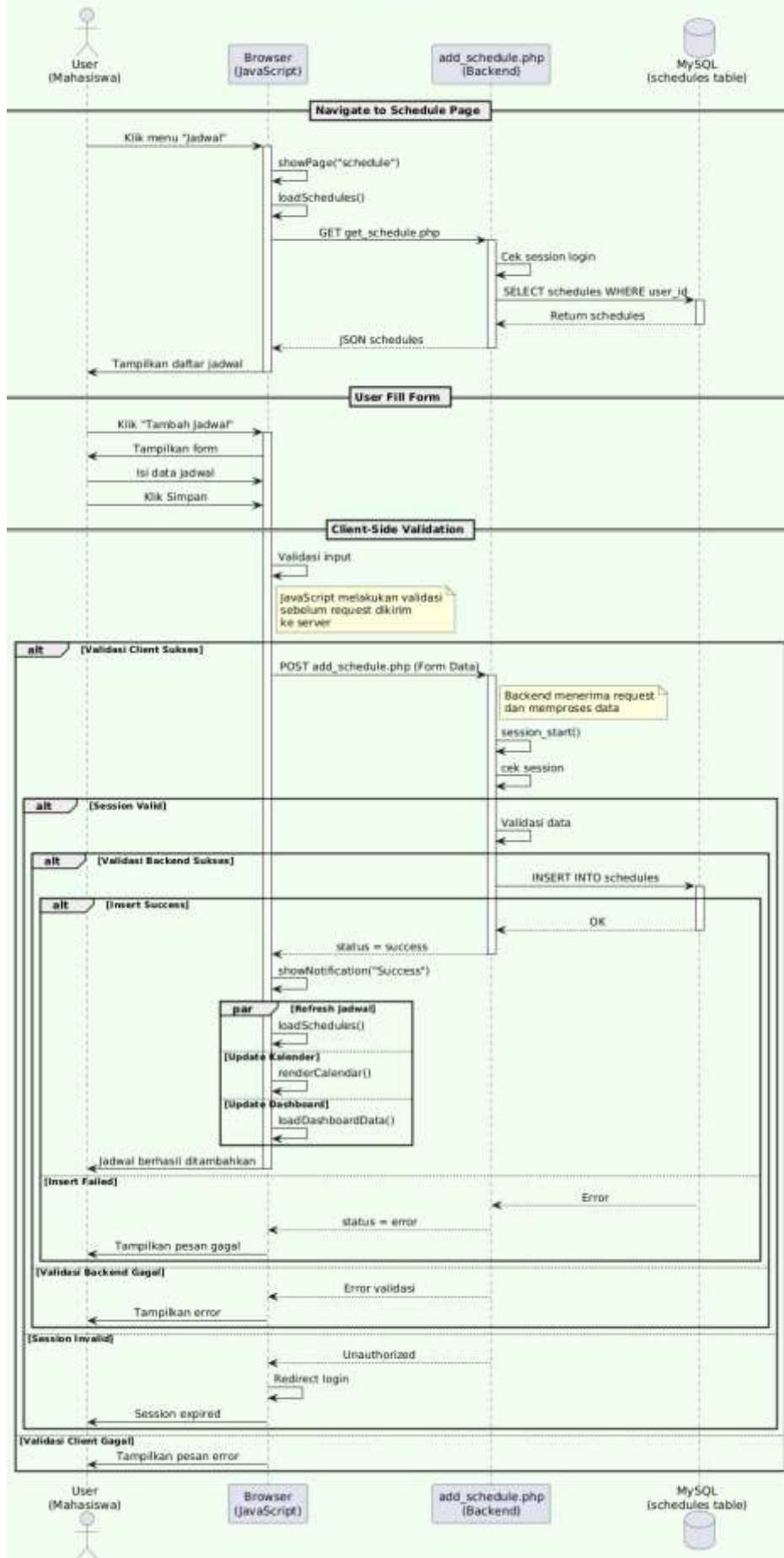
Activity Diagram - Dashboard & Calendar ARMO System



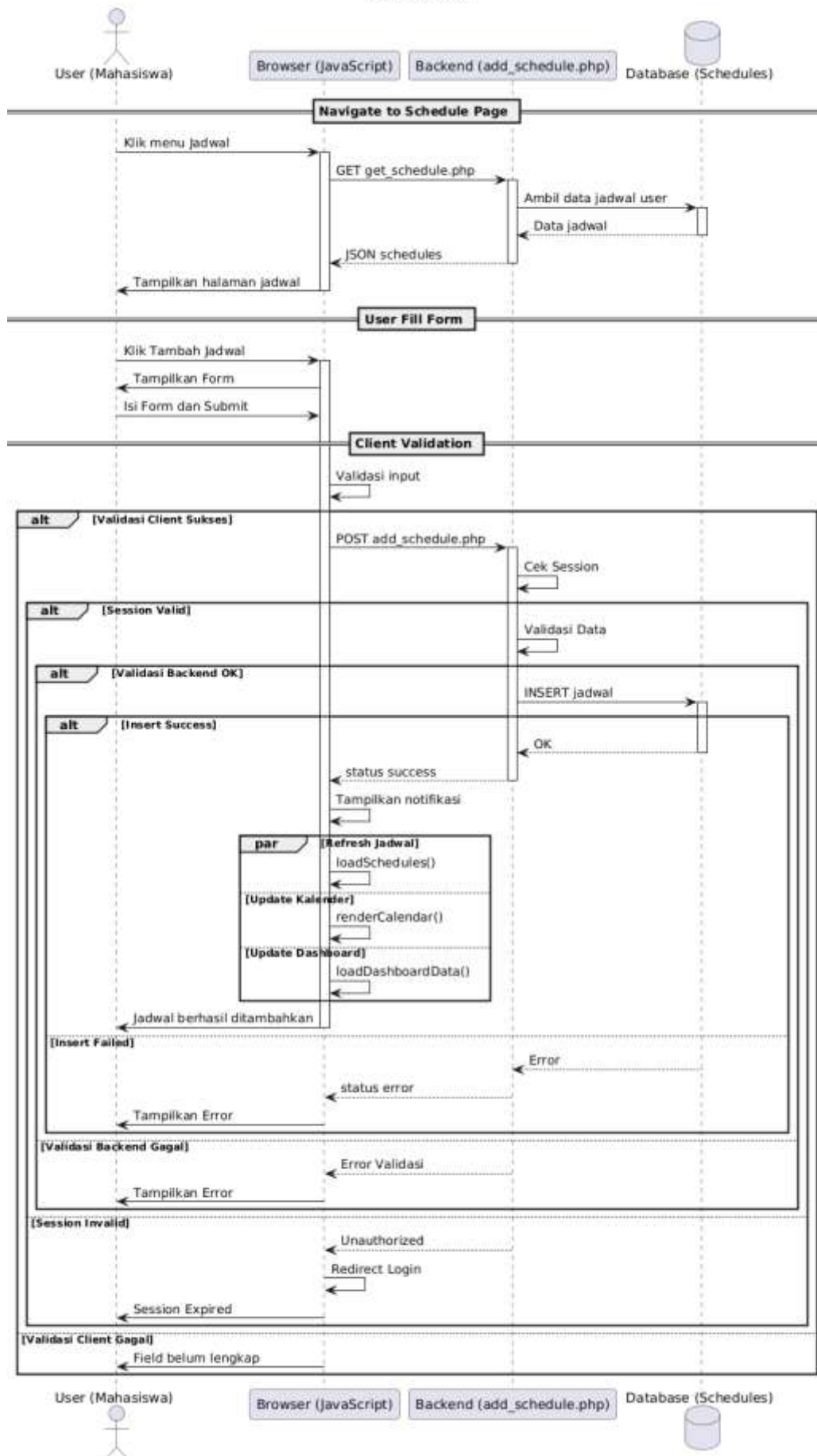
2.7.5. Sequence Diagram



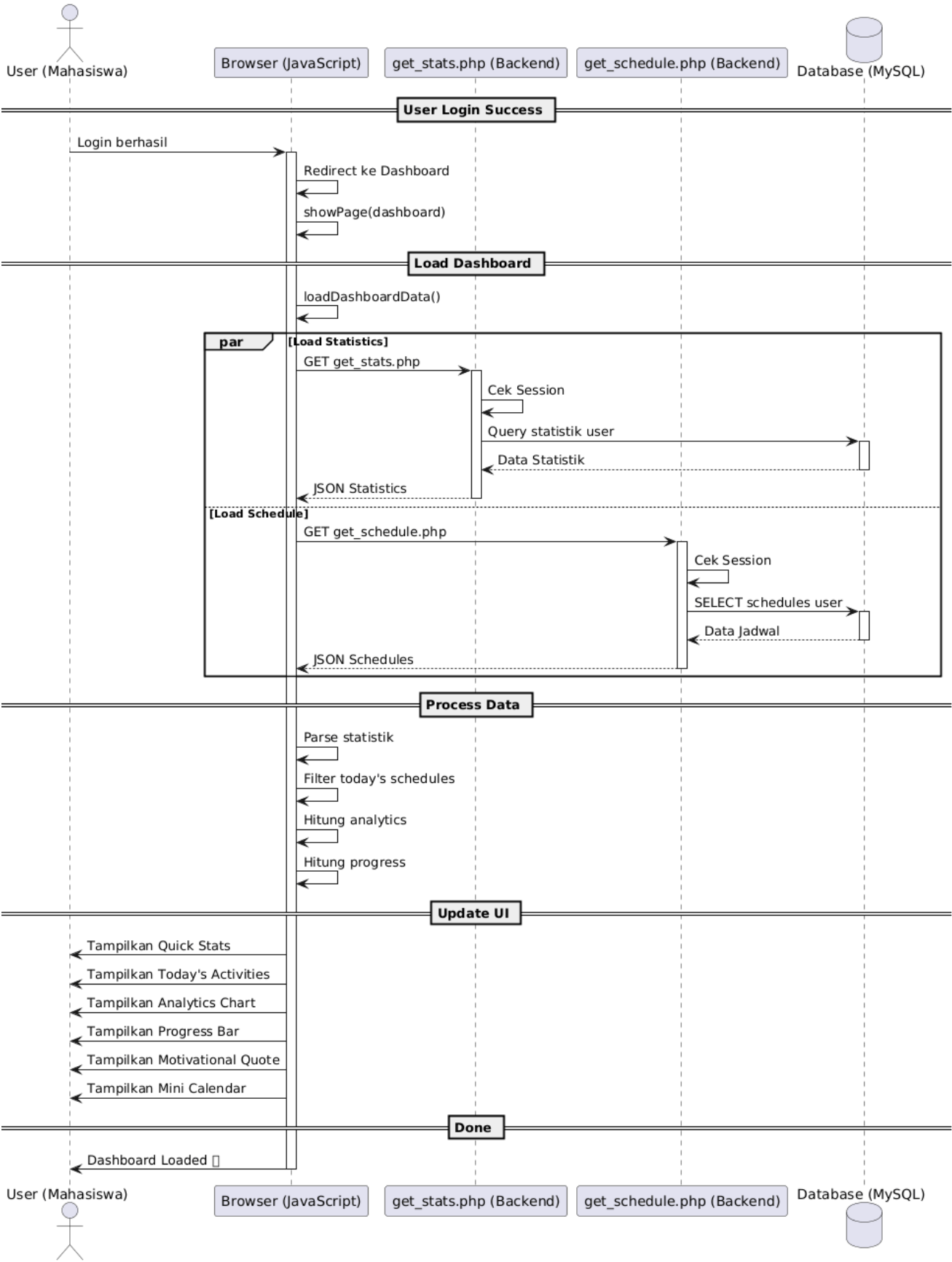
**Sequence Diagram - Tambah Jadwal (Schedule)
ARVO System**



Sequence Diagram - Tambah Jadwal (Schedule) ARVO System



Sequence Diagram - Load Dashboard & Statistics
ARVO System



BAB III

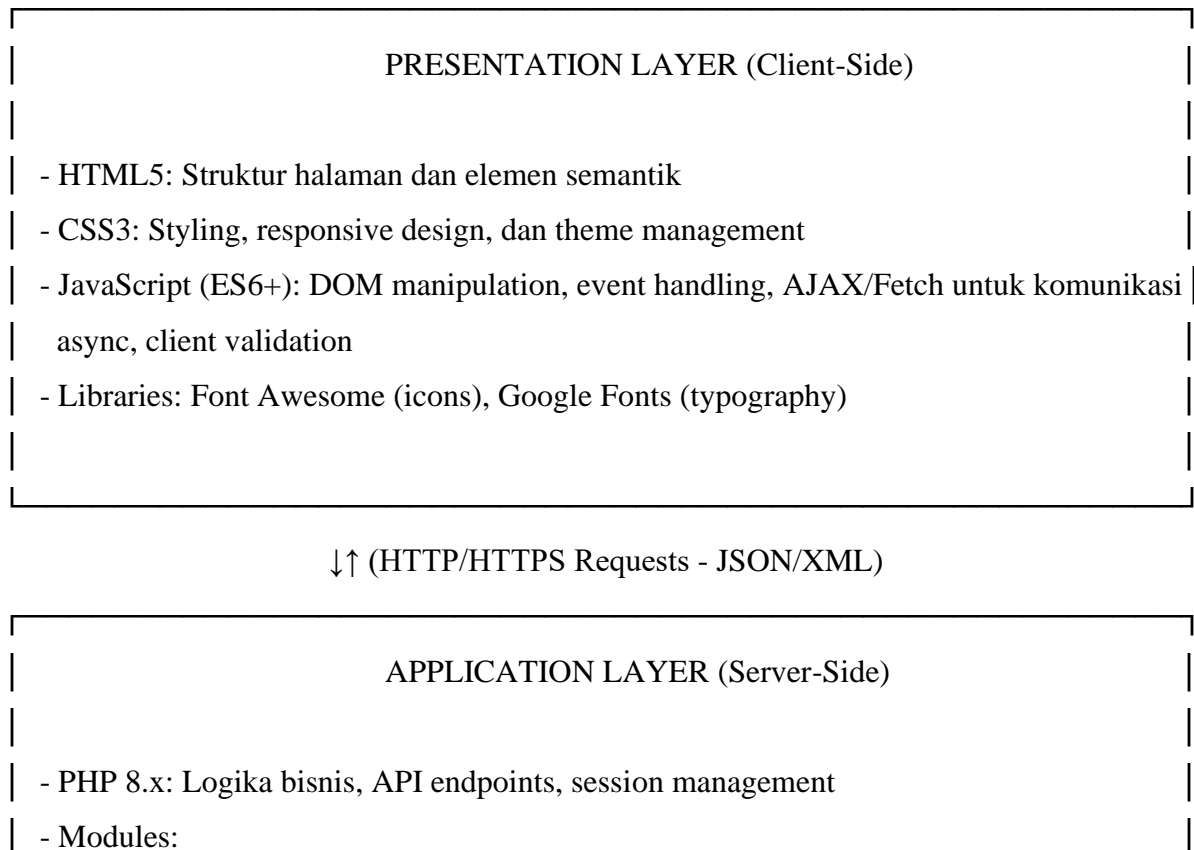
PERANCANGAN SISTEM

3.1 Arsitektur Sistem

Arsitektur sistem ARVO dirancang dengan prinsip skalabilitas, modularitas, dan pemisahan tanggung jawab (separation of concerns) untuk memastikan kemudahan pemeliharaan dan pengembangan lebih lanjut. Pendekatan ini mengikuti model arsitektur 3-tier (three-tier architecture) yang umum digunakan dalam aplikasi web, sebagaimana dijelaskan oleh IBM (2023) sebagai struktur yang memisahkan lapisan presentasi, aplikasi, dan data untuk meningkatkan keamanan dan performa. Model ini dipilih karena sesuai dengan kebutuhan aplikasi berbasis web yang memerlukan interaksi dinamis antara client dan server, sambil menjaga isolasi data sensitif.

3.1.1. Arsitektur Umum

ARVO mengadopsi arsitektur client-server dengan tiga lapisan utama, yaitu presentation tier (UI), application tier (logika bisnis), dan data tier (penyimpanan). Berikut adalah representasi diagramatis:



• Auth: Login/Register/Logout
• Schedule: CRUD operations for schedules
• Notes: CRUD for notes
• Goals: CRUD for goals with progress tracking
• Profile: User data management
• Stats: Aggregation queries for dashboard
- Security: Input sanitization, prepared statements
- Web Server: Apache (via XAMPP for development)

↓↑ (SQL Queries - Prepared Statements)

APPLICATION LAYER (Server-Side)
- MySQL 8.0+: Relational DBMS
- Tables:
• users (authentication and profile)
• schedules (scheduling data)
• notes (notetaking)
• goals (goal tracking)
• activity (user activity logs)
- Relationships: 1:M from users to other tables
- Indexing: For optimized queries on frequent filters

Diagram ini mengilustrasikan alur data dari client ke server dan database, memastikan pemisahan yang jelas untuk menghindari ketergantungan langsung antara UI dan data, untuk arsitektur hemat biaya dan skalabel.

3.1.2. Komponen Sistem

- Frontend (Presentation Layer):
 Bertanggung jawab atas interaksi pengguna. Komponen ini dirancang untuk responsif, dengan dukungan untuk perangkat mobile hingga desktop. JavaScript menangani validasi klien-side untuk mengurangi beban server, sementara Fetch API digunakan untuk komunikasi asinkron dengan backend.

2. Backend (Application Layer):

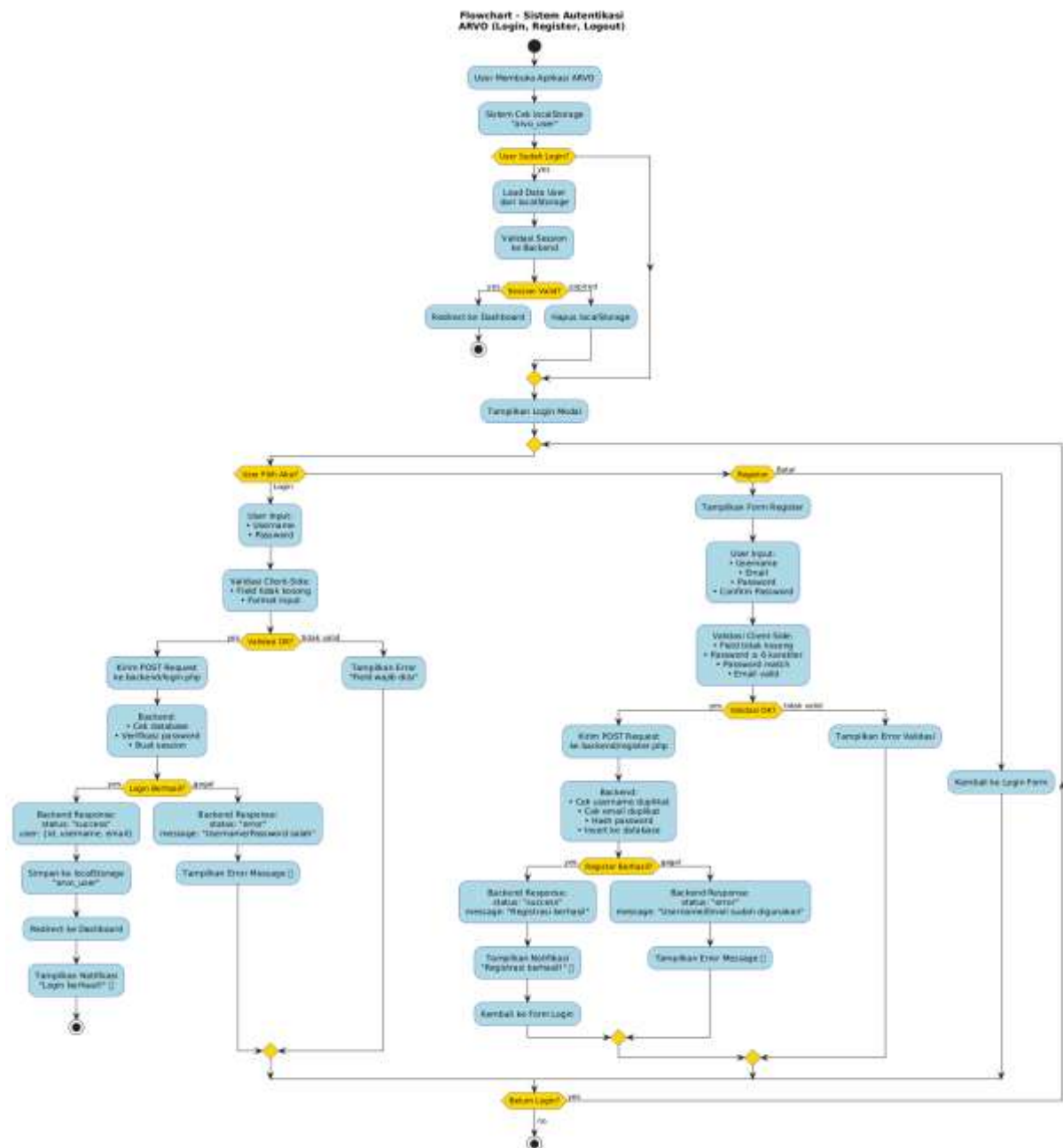
PHP memproses logika bisnis, termasuk validasi server-side dan operasi CRUD. Endpoints dirancang seperti RESTful API (misalnya, `/backend/schedule/add_schedule.php`) untuk modularitas. Session management PHP native digunakan untuk autentikasi, dengan regenerasi session untuk keamanan.

3. Database (Data Layer):

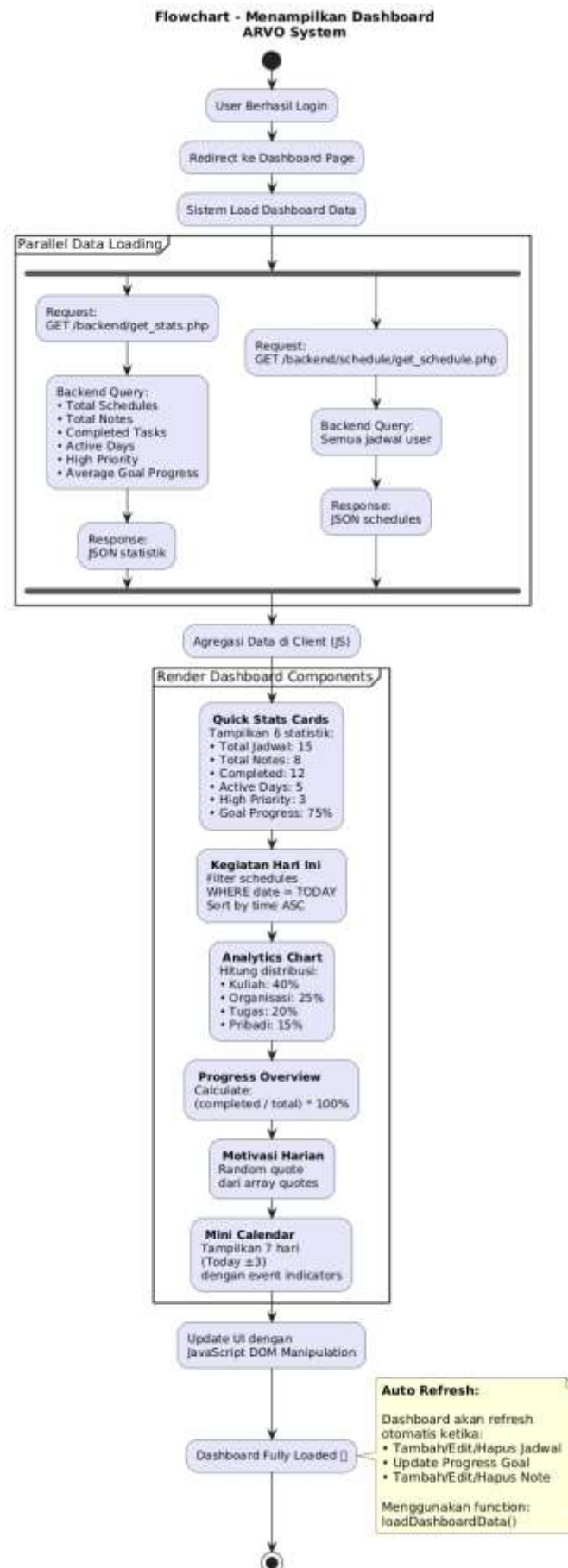
MySQL dipilih karena kemampuannya dalam menangani data relasional dengan efisiensi tinggi. Tabel-tabel dirancang dengan foreign keys untuk integritas referensial.

3.1.3. Alur Kerja Sistem

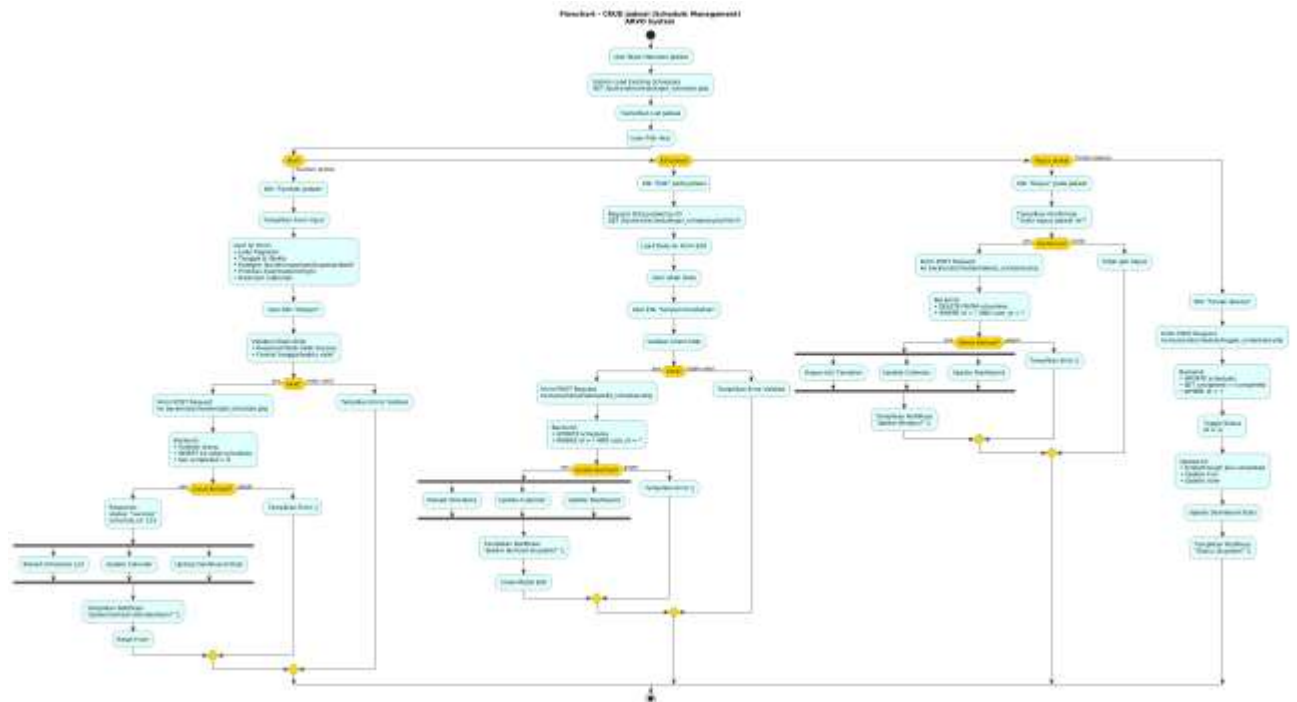
1. Alur Autentikasi



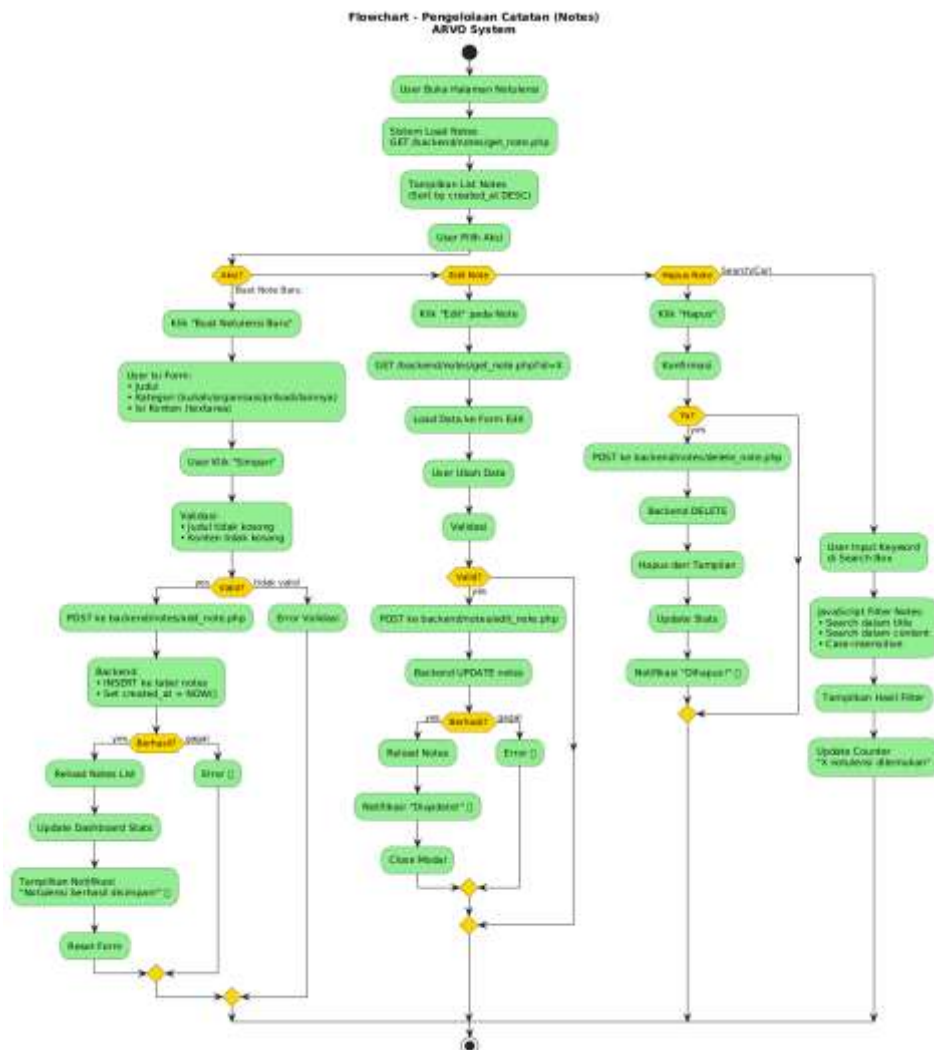
2. Alur Menampilkan Dashboard



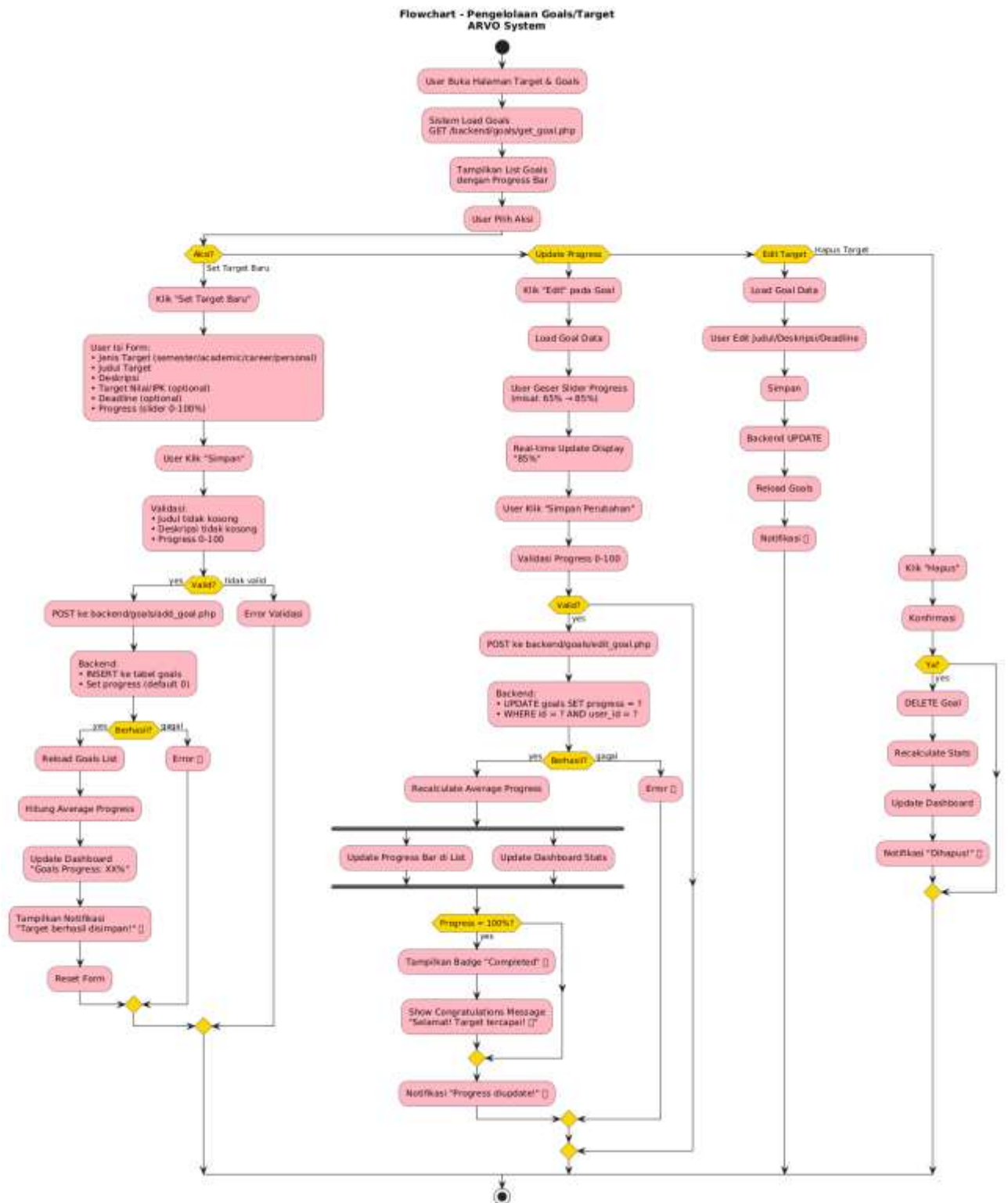
3. Alur CRUD Jadwal



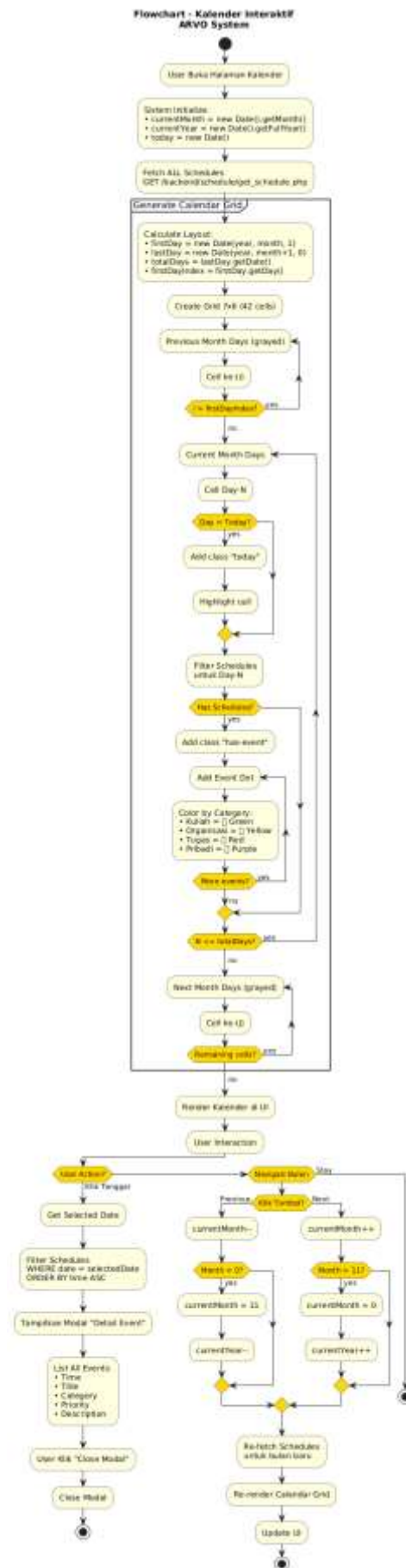
4. Alur Pengelolaan Catatan



5. Alur Pengelolaan Target

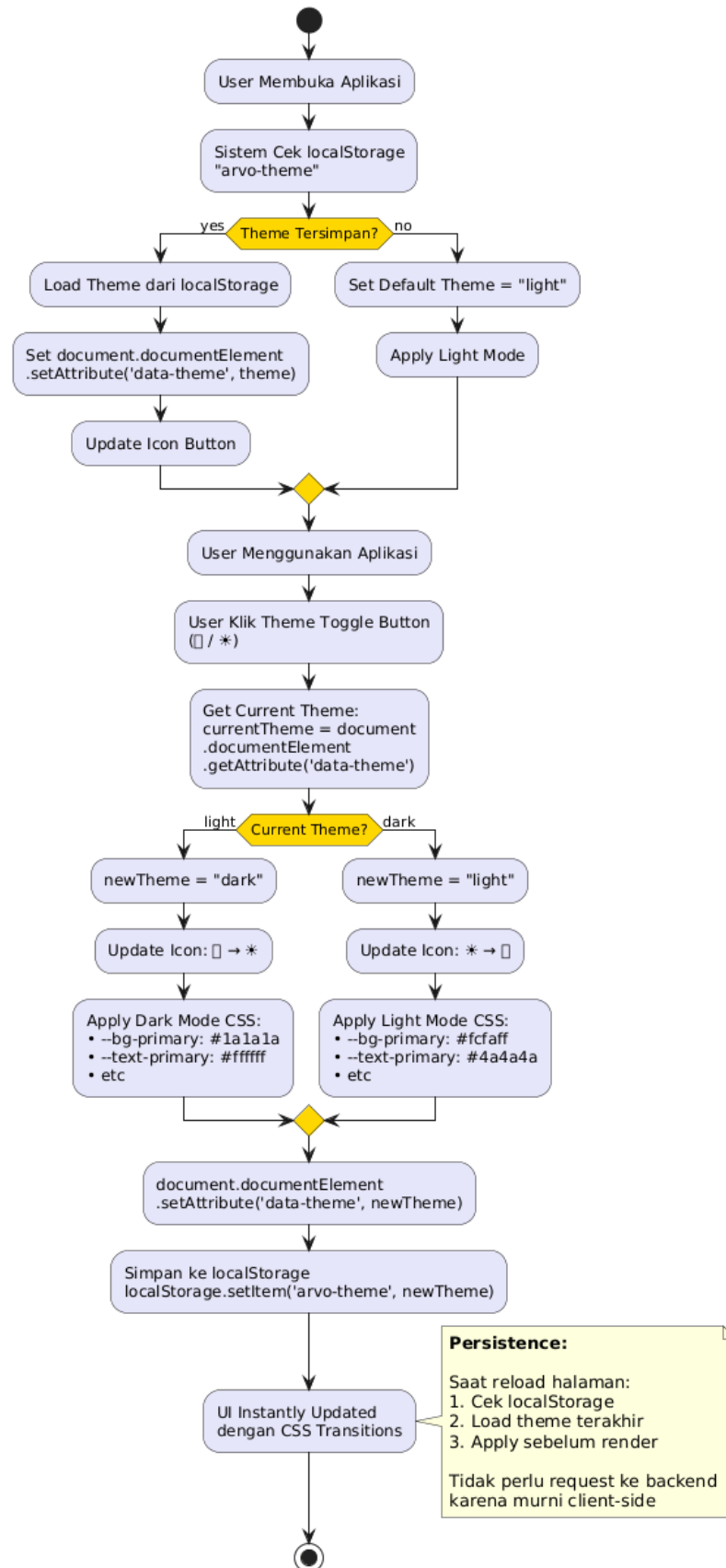


6. Alur Kalender

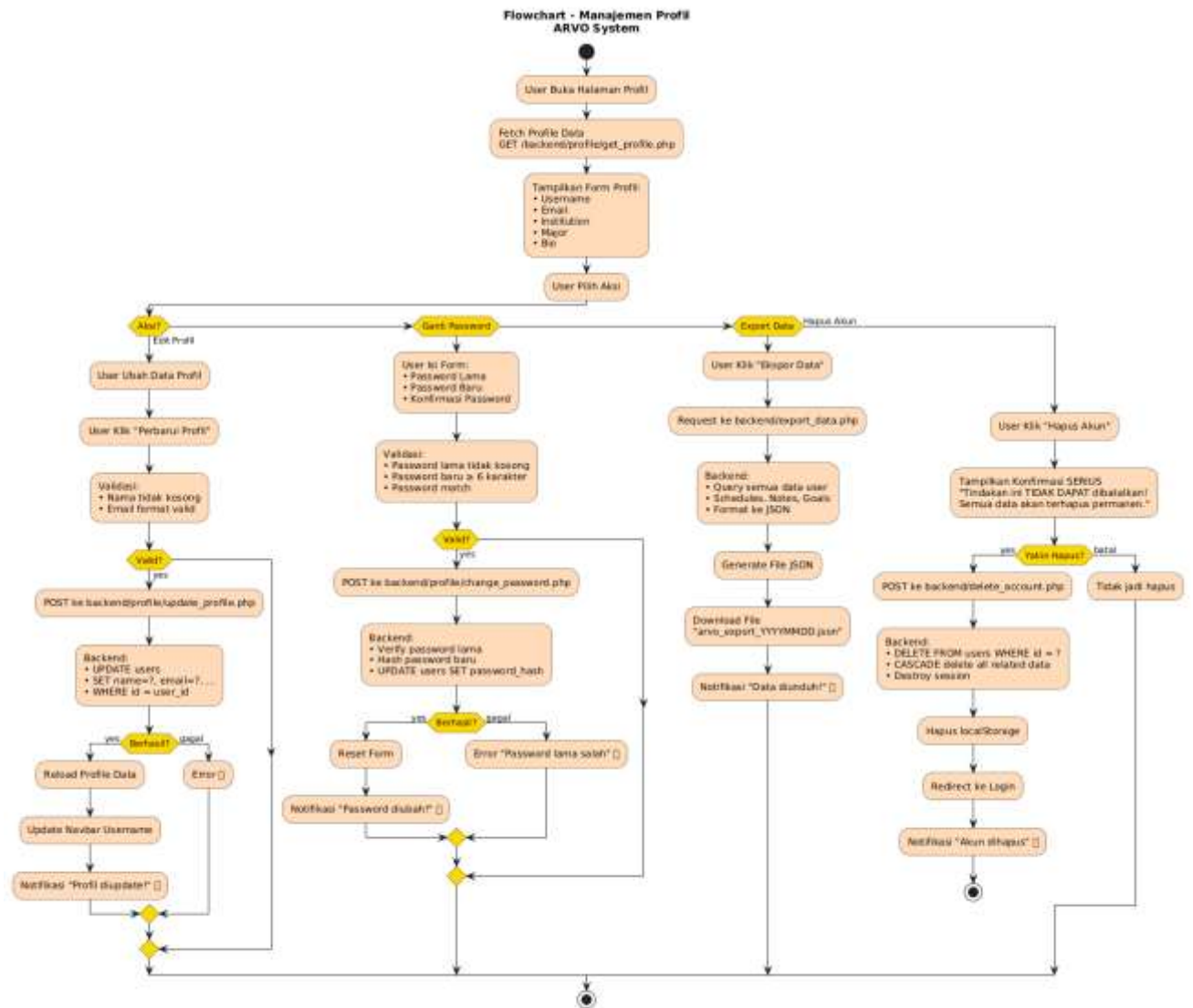


7. Alur Theme Mode

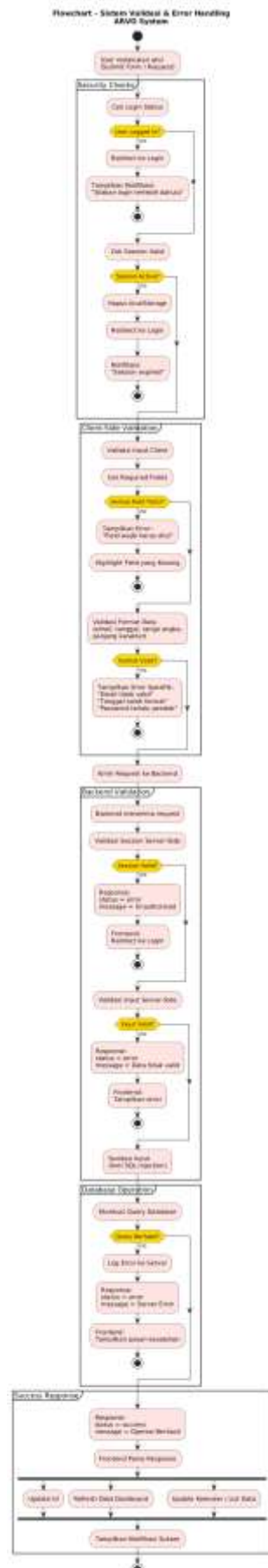
Flowchart - Theme Toggle (Light/Dark Mode)
ARVO System



8. Alur Manajemen Profil



9. Alur Validasi & Error Handling



3.1.4. Security Layer

Keamanan diimplementasikan multi-layer:

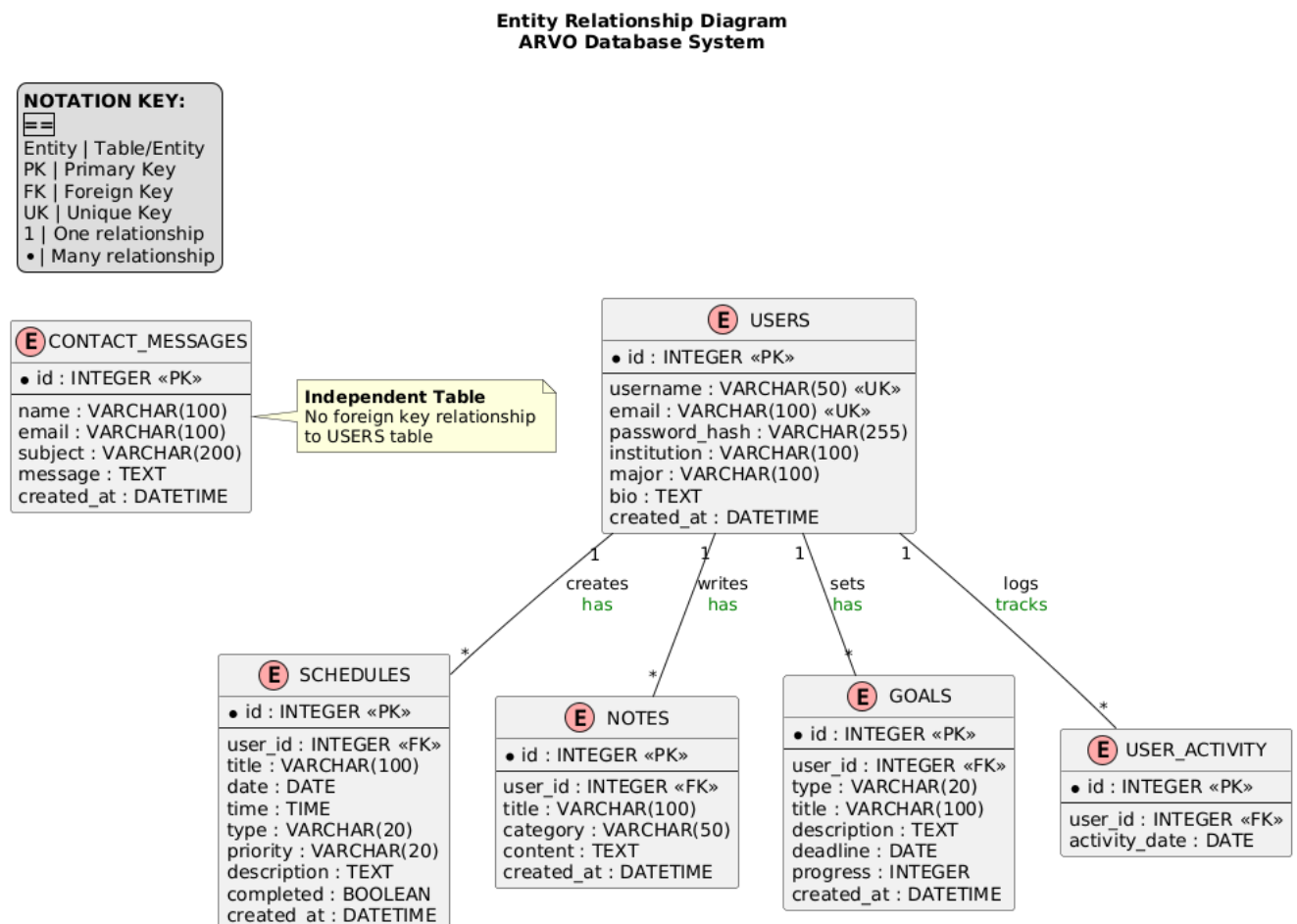
- Input Security: Validasi klien dan server; prepared statements untuk mencegah SQL injection.
- Session Security: Timeout 30 menit, regenerasi ID session, secure cookies.
- Access Control: Semua query difilter oleh user_id untuk mencegah unauthorized access.

3.2 Desain Database

Desain database ARVO mengikuti prinsip normalisasi untuk mengurangi redundansi dan memastikan integritas data. Database dinormalisasi hingga 3NF untuk menghindari anomali update.

3.2.1. Entity Relationship Diagram (ERD)

Perancangan database dilakukan dengan mempertimbangkan efisiensi kueri dan integritas referensial.



3.2.2. Struktur Tabel

1. Tabel: users

Nama Kolom	Tipe Data	Atribut	Keterangan
id	INT	PK, AI	ID unik pengguna
username	VARCHAR(50)	UNIQUE, NOT NULL	Nama pengguna untuk login
email	VARCHAR(100)	UNIQUE, NOT NULL	Alamat email pengguna
password_hash	VARCHAR(255)	NOT NULL	Password yang sudah dienkripsi
institution	VARCHAR(150)	-	Nama universitas/instansi
major	VARCHAR(150)	-	Jurusan/program studi
bio	TEXT	-	Deskripsi singkat profil
created_at	TIMESTAMP	DEFAULT NOW	Waktu pendaftaran akun

2. Tabel: schedules

Nama Kolom	Tipe Data	Atribut	Keterangan
id	INT	PK, AI	ID unik jadwal
user_id	INT	FK, NOT NULL	Relasi ke users.id
title	VARCHAR(150)	NOT NULL	Nama kegiatan
date	DATE	NOT NULL	Tanggal pelaksanaan
time	TIME	NOT NULL	Jam pelaksanaan
type	ENUM	NOT NULL	Kategori (kuliah, organisasi, dll)
priority	ENUM	NOT NULL	Tingkat urgensi (low, medium, high)
description	TEXT	-	Detail keterangan jadwal
completed	TINYINT(1)	DEFAULT 0	Status selesai (0: Belum, 1: Sudah)
created_at	TIMESTAMP	DEFAULT NOW	Waktu data dibuat

3. Tabel: notes

Nama Kolom	Tipe Data	Atribut	Keterangan
id	INT	PK, AI	ID unik catatan
user_id	INT	FK, NOT NULL	Relasi ke users.id
title	VARCHAR(150)	NOT NULL	Judul notulensi
category	ENUM	NOT NULL	Kategori (kuliah, organisasi, dll)
content	TEXT	NOT NULL	Isi teks catatan
created_at	TIMESTAMP	DEFAULT NOW	Waktu catatan dibuat

4. Tabel: goals

Nama Kolom	Tipe Data	Atribut	Keterangan
id	INT	PK, AI	ID unik target
user_id	INT	FK, NOT NULL	Relasi ke users.id
type	ENUM	NOT NULL	Jenis target (semester, karir, dll)
title	VARCHAR(150)	NOT NULL	Nama target yang ingin dicapai
description	TEXT	-	Penjelasan detail target
deadline	DATE	-	Batas waktu pencapaian
progress	INT	DEFAULT 0	Persentase progres (0-100)
created_at	TIMESTAMP	DEFAULT NOW	Waktu target dibuat

5. Tabel: user_activity

Nama Kolom	Tipe Data	Atribut	Keterangan
id	INT	PK, AI	ID unik log
user_id	INT	FK, NOT NULL	Relasi ke users.id
activity_date	DATE	UNIQUE KEY*	Tanggal pengguna aktif

6. Tabel: contact_messages

Nama Kolom	Tipe Data	Atribut	Keterangan
id	INT	PK, AI	ID unik pesan
name	VARCHAR(150)	NOT NULL	Nama pengirim pesan
email	VARCHAR(150)	NOT NULL	Email pengirim pesan
subject	VARCHAR(150)	-	Perihal pesan
message	TEXT	NOT NULL	Isi pesan lengkap
created_at	TIMESTAMP	DEFAULT NOW	Waktu pesan dikirim

➤ Relasi Antar-Tabel:

a) Tipe Relasi

- users → schedules: One-to-Many (1:M)
- users → notes: One-to-Many (1:M)
- users → goals: One-to-Many (1:M)
- users → activity: One-to-Many (1:M)

b) Aturan Cascade

- ON DELETE CASCADE: Jika user dihapus, semua data terkait ikut terhapus
- ON UPDATE CASCADE: Jika user_id berubah, foreign key otomatis update

3.3 Perancangan Tampilan

Perancangan antarmuka (UI/UX) pada aplikasi ARVO – Aplikasi Manajemen Akademik Mahasiswa dilakukan agar sistem tidak hanya berfungsi dengan baik, tetapi juga memberikan pengalaman penggunaan yang nyaman, mudah dipahami, dan menarik secara visual. Desain UI mengutamakan kesederhanaan, konsistensi, dan kemudahan navigasi sehingga mahasiswa sebagai pengguna utama dapat menggunakan aplikasi tanpa kesulitan.

3.3.1 Konsep Desain dan Pendekatan UI/UX

Antarmuka ARVO dirancang menggunakan pendekatan modern, clean, dan minimalis dengan memanfaatkan desain dashboard interaktif berbasis web. Setiap elemen disusun secara terstruktur agar informasi akademik dapat disajikan secara ringkas namun tetap informatif. Penggunaan ikon Font Awesome, tipografi Poppins, serta warna-warna lembut (pastel) memberikan kesan ramah, tidak kaku, dan sesuai karakter mahasiswa.

Secara UX, sistem dirancang berbasis konsep:

- simple interaction → interaksi sederhana dan mudah dipahami
- consistency → gaya tampilan seragam di seluruh halaman
- focus on readability → teks jelas dan mudah dibaca
- productivity support → UI membantu mahasiswa fokus pada aktivitasnya

3.3.2 Prinsip UI/UX yang Diterapkan

Beberapa prinsip desain yang diterapkan pada ARVO antara lain:

1. User Friendly
Navigasi dirancang jelas melalui navbar utama, menu yang terstruktur, serta ikon yang mudah dikenali.
2. Consistency
Warna, font, ikon, dan layout ditetapkan konsisten berdasarkan komponen CSS global.
3. Responsive Design
Website dilengkapi hamburger menu dan layout yang menyesuaikan ukuran layar sehingga nyaman digunakan di laptop maupun mobile.
4. Clarity & Readability
Menggunakan font Poppins dengan kombinasi heading tebal dan konten ringan agar teks mudah dipahami.
5. Feedback Interaction
Sistem memberikan respons visual seperti hover effect, highlight tombol, dan notifikasi sukses/gagal.

3.3.3 Struktur Navigasi Sistem

Sistem menggunakan navbar sebagai pusat navigasi utama. Setelah login, pengguna dapat mengakses beberapa halaman berikut:

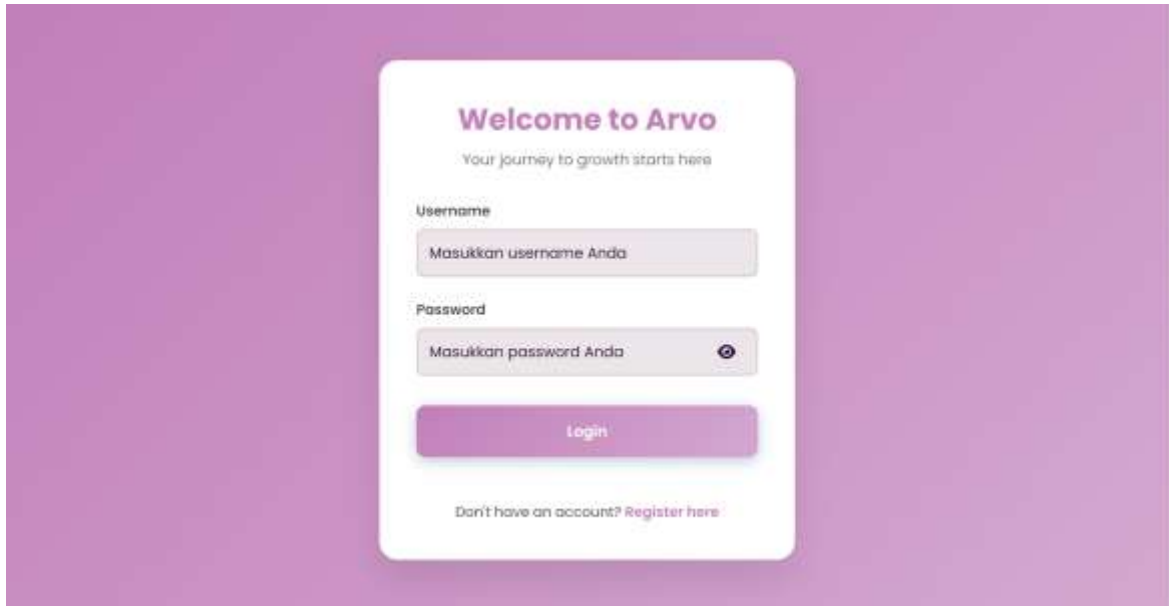
- Dashboard
- Schedule (Jadwal)
- Notes (Catatan)
- Goals (Target)
- Calendar
- Profile
- Settings

Navigasi dirancang menggunakan menu atas (top navigation bar) dengan dukungan hamburger menu untuk perangkat mobile.

3.3.4 Desain Setiap Halaman Antarmuka

A. Halaman Login

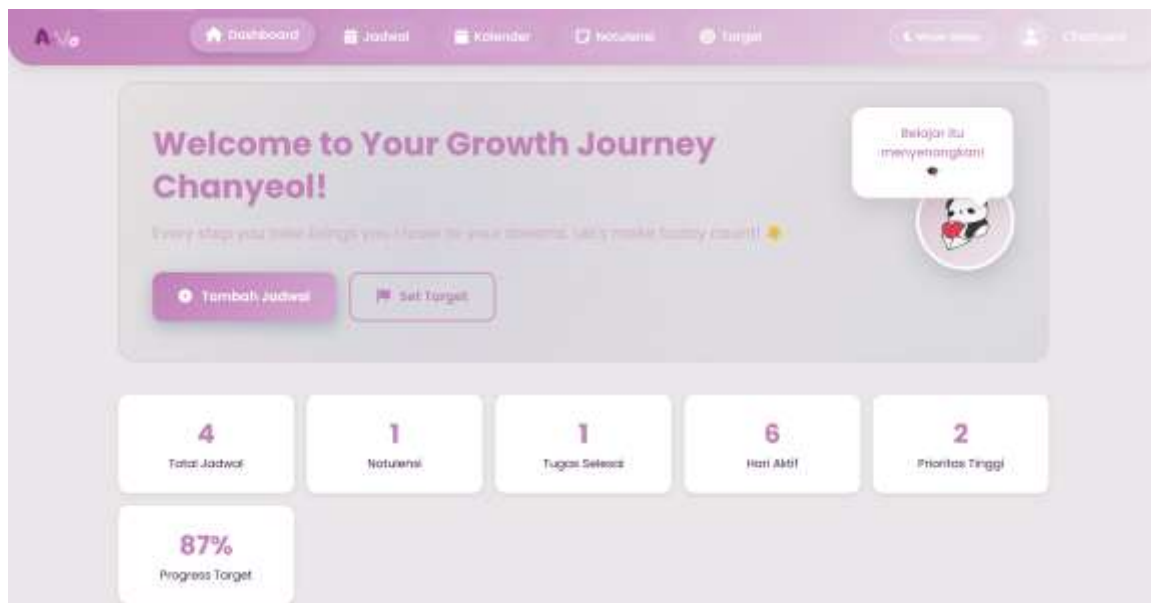
Antarmuka login ditampilkan sebagai modal fullscreen dengan latar gradient pastel.



Desain ini dipilih agar pengguna langsung fokus pada proses autentikasi sebelum masuk ke sistem.

B. Dashboard

Dashboard merupakan halaman utama setelah pengguna berhasil login. Tampilan terdiri dari:

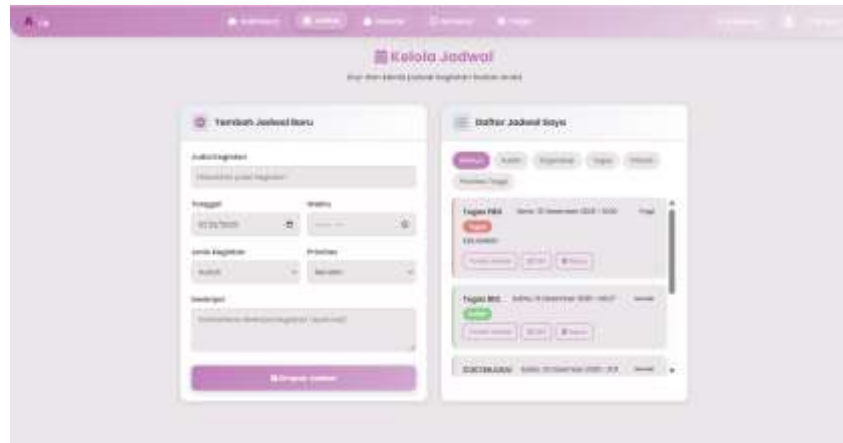


Dashboard dirancang informatif, membantu mahasiswa melihat gambaran umum aktivitas akademiknya secara cepat.

C. Halaman Schedule

Pada halaman ini pengguna dapat:

- melihat daftar jadwal
- menambahkan jadwal
- mengedit kegiatan
- menghapus jadwal
- menandai aktivitas selesai



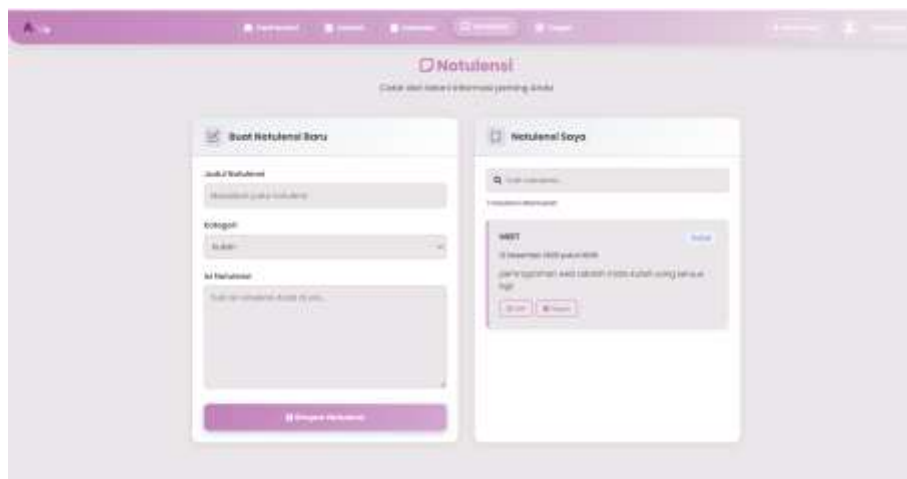
Setiap jadwal dilengkapi kategori warna seperti:

- hijau untuk kegiatan kuliah
- kuning untuk organisasi
- merah untuk tugas
- ungu untuk aktivitas pribadi

Hal ini membantu visualisasi prioritas kegiatan.

D. Halaman Notulensi

Halaman ini digunakan untuk menyimpan catatan penting mahasiswa.



Desain dibuat sederhana seperti “note board” agar natural seperti mencatat manual.

E. Halaman Target

Halaman ini menampilkan:

The screenshot displays the 'Target & Karir' application interface. The top navigation bar includes links for Dashboard, Jadwal, Kalender, Akademik, and Target. The main content area is divided into two panels. The left panel, titled 'Set Target Baru', contains a form for creating a new target with fields for 'Jenis Target' (Target Semester), 'Jadwal Target' (Mau lakukan jadwal target), 'Deskripsi' (Jelaskan target Anda...), 'Target Nilai/IPK' (Contoh 3.0), 'Tanggal Waktu' (12/23/2025), and a 'Progress Saat ini (%)' bar at 0%. A 'Simpan target' button is at the bottom. The right panel, titled 'Daftar Target Saya', shows a 'MENUJU SUKSES' section with 'Rajih Soedjor' as the user, 'Target Waktu: 31 Desember 2025', and a 'Progress: 87%' bar. It includes 'Kiri' and 'Kanan' buttons.

Penggunaan progress bar membantu mahasiswa memantau perkembangan target akademiknya.

F. Halaman Calendar

Halaman kalender bersifat interaktif dan terintegrasi dengan jadwal.



G. Halaman Profile & Settings

Halaman ini menampilkan:

Profil Pengguna
Kelola informasi profil dan pengaturan akun Anda

Informasi Profil

Nama lengkap
Charyesi

Email
charyesi@gmail.com

Instansi
Nama Instansi

Jurusan
Jurusan pendidikan

Bio
Ceritakan sedikit tentang diri Anda...

Pengaturan Akun

Ubah Password

Password saat ini

Password Baru

Konfirmasi Password Baru

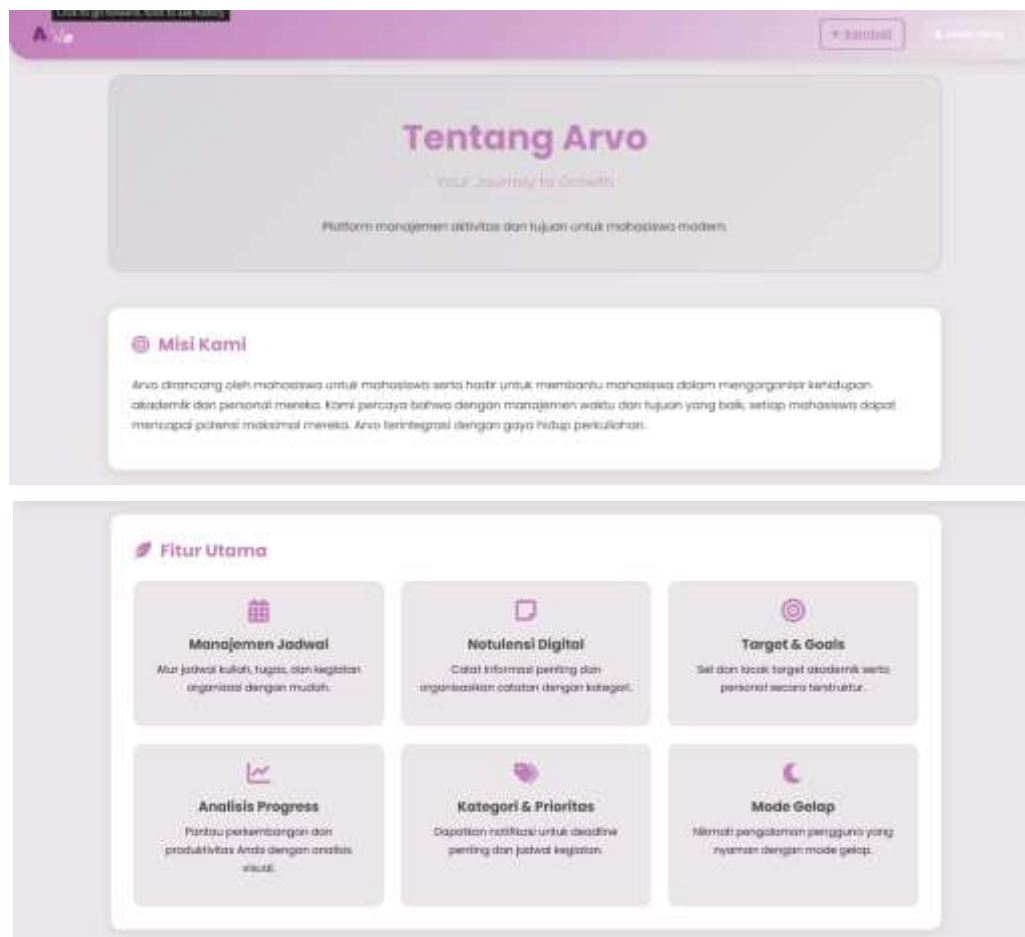
Privasi & Data

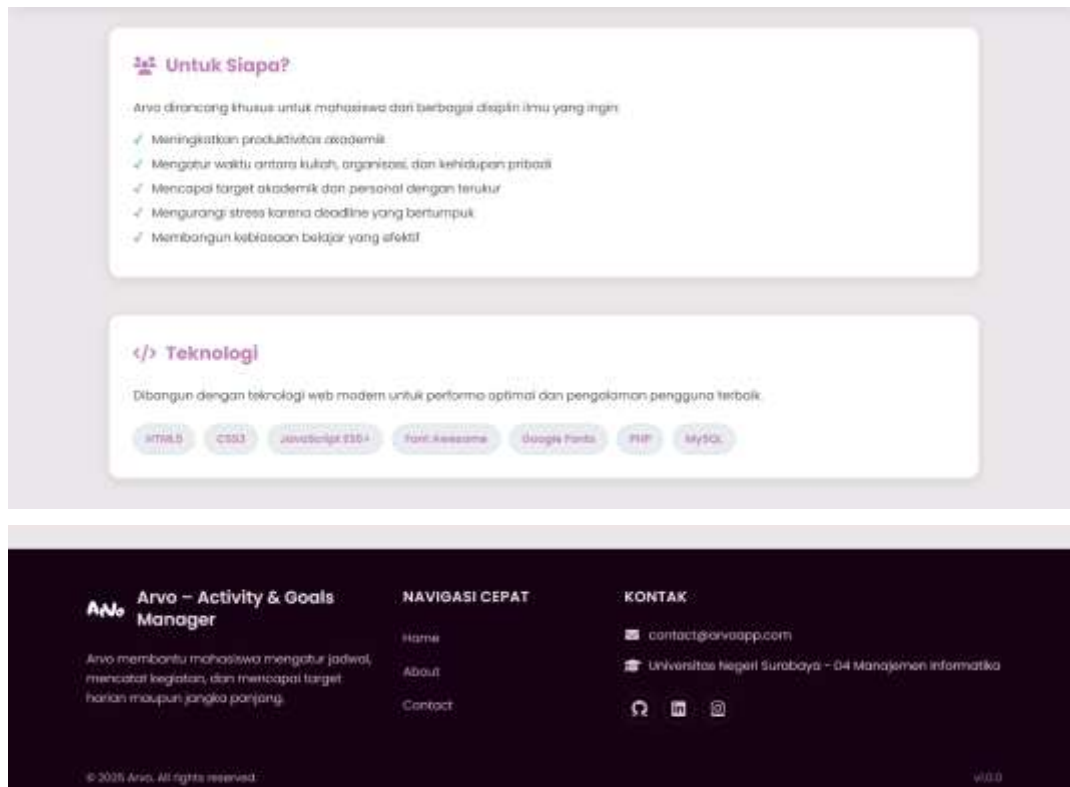
Hapus Data

Hapus Akun

H. Halaman About

Halaman ini menampilkan:





I. Halaman Contact

Halaman ini menampilkan:

Hubungi Kami

Ada beberapa cara untuk menghubungi Arvo (Email, Form, atau Telepon):

Form Kontak

Isi form berikut untuk mengirim pesan kepada tim Arvo:

Nama Lengkap *

Isikan nama lengkap Anda

Harus diisi nama sesuai dengan identitas Anda

Email *

Harus diisi email yang valid untuk menerima pesan Anda

Telepon (Opsional)

Harus diisi nomor telepon yang valid

Isi Pesan *

Isi pesan yang ingin Anda sampaikan

Lampiran (Opsional)

Anda dapat menambahkan screenshot (jpg, png) atau dokumen (pdf)

☐ Saya setuju bahwa pesan ini akan diproses [Kebijakan Privasi](#)

Informasi Kontak

Email

contact@arvoapp.com

Batas waktu respon: dalam 24 jam

Alamat

Universitas Jegeri Surabaya
D4 Manajemen Informatika

Pengembang

Tim Pengembang Arvo
Anggota: Arif Nur Hafid

Pertanyaan Umum

Kapan saya akan menerima balasan?

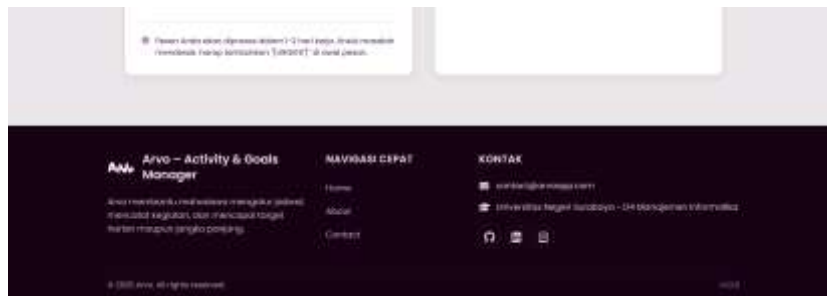
Tim Arvo akan membalas pesan dalam 1-3 hari kerja. Pesan dengan prioritas tinggi akan diproses lebih cepat.

Apakah saya mendapatkan file?

Tim Arvo dapat membantu Anda dengan cara lain jika diperlukan dengan format jpg, png, pdf, atau lainnya.

Bagaimana cara melaporkan bug?

Harap kirimkan 'bug-report' dan lampirkan screenshot jika memungkinkan untuk membantu kami memperbaiki masalah.



3.3.5 Desain Visual

Aplikasi ARVO menggunakan kombinasi warna bernuansa lembut (soft pastel) seperti:

Elemen	Warna
Primary (Soft Blue / Pink)	#c27fb9 – #d4a8d1
Secondary	rgb(235, 230, 233)
Accent	#db88a4
Background	#fcfaff
Text Primary	#4a4a4a
Text Secondary	#31113a

Warna ini dipilih untuk memberikan kesan:

- nyaman dilihat
- tidak melelahkan mata
- cocok untuk nuansa akademik mahasiswa

Typography:

- Font Utama: Poppins
- Beban font variatif (300–700) agar fleksibel
- Ikon menggunakan Font Awesome

3.3.6 Desain Responsif

ARVO mendukung tampilan responsive dengan:

- penggunaan flexbox dan grid
- navbar yang berubah menjadi hamburger saat layar kecil
- layout konten menyesuaikan ukuran device
- komponen UI tetap rapi di mobile

Dengan demikian aplikasi tetap nyaman digunakan di berbagai perangkat.

3.3.7 Alur Pengalaman Pengguna Secara Singkat

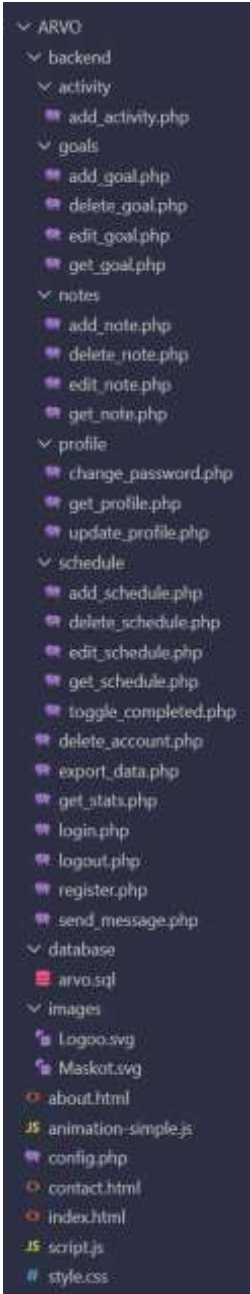
1. Pengguna membuka website
2. Pengguna login melalui login modal
3. Sistem mengarahkan ke dashboard
4. Pengguna dapat mengakses jadwal, catatan, target, kalender, profil, dll.
5. Sistem menyimpan perubahan dan memperbarui tampilan secara dinamis
6. Pengguna dapat keluar (logout) dan sistem menutup session

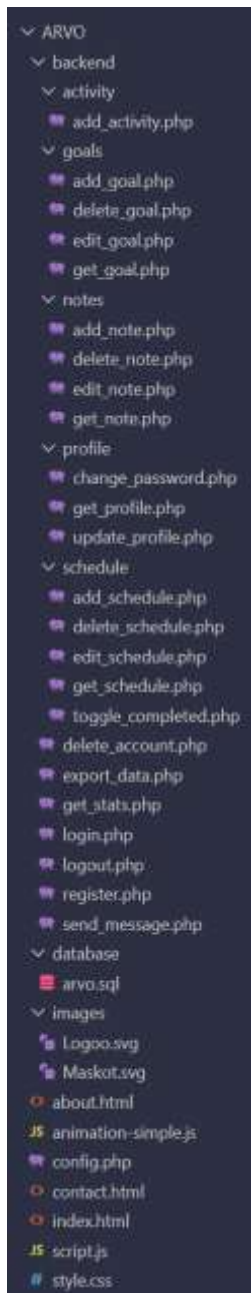
BAB IV

IMPLEMENTASI WEBSITE

4.1 Struktur Folder

Struktur folder digunakan untuk mengorganisasi semua komponen website ARVO agar mudah dipelihara, dibaca, dan diperbarui. Seluruh file diletakkan di dalam satu direktori utama bernama `/arvo` seperti berikut:

File	Penjelasan
	<p>Direktori Utama</p> <p>Direktori akar berisi file-file utama yang mengatur tampilan dasar dan konfigurasi global aplikasi:</p> <ul style="list-style-type: none">• <code>index.html</code>: Halaman utama aplikasi yang berfungsi sebagai entry point bagi pengguna.• <code>about.html</code>: Halaman yang berisi informasi mengenai profil aplikasi ARVO.• <code>contact.html</code>: Halaman formulir kontak untuk komunikasi pengguna dengan pengembang.• <code>config.php</code>: File konfigurasi sentral yang mengatur koneksi antara aplikasi PHP dengan database MySQL.• <code>script.js</code>: File utama untuk logika frontend dan interaktivitas menggunakan Vanilla JavaScript.• <code>style.css</code>: Dokumen stylesheet tunggal yang mengatur seluruh estetika, tata letak, dan responsivitas aplikasi.• <code>animation-simple.js</code>: Skrip khusus yang menangani efek visual dan transisi antar elemen agar antarmuka terasa lebih dinamis. <p>Direktori backend/</p> <p>Folder ini merupakan inti dari logika bisnis aplikasi yang terbagi menjadi beberapa sub-modul berdasarkan fitur utama:</p> <ol style="list-style-type: none">1. Modul Fitur:<ul style="list-style-type: none">• <code>activity/</code>: Berisi <code>add_activity.php</code> untuk mencatat log aktivitas harian pengguna guna keperluan statistik.• <code>goals/</code>: Mengelola fungsionalitas target, mencakup



pembuatan (add), penghapusan (delete), penyuntingan (edit), dan pengambilan data (get) target akademik atau pribadi.

- notes/: Menangani sistem notulensi digital dengan operasi CRUD (Create, Read, Update, Delete) lengkap melalui file add, delete, edit, dan get_note.php.
- profile/: Mengatur data akun pengguna, termasuk fitur change_password.php, get_profile.php, dan update_profile.php.
- schedule/: Modul paling kompleks yang mengelola jadwal harian, dilengkapi file tambahan toggle_completed.php untuk mengubah status penyelesaian tugas.

2. Core Backend Services:

- login.php & logout.php: Menangani manajemen sesi dan autentikasi keamanan pengguna.
- register.php: Mengatur pendaftaran akun pengguna baru ke dalam sistem.
- get_stats.php: Menghasilkan data agregat untuk ditampilkan pada grafik analitik di dashboard.
- export_data.php: Fitur untuk mengunduh data pengguna dalam format terstruktur.
- delete_account.php: Prosedur penghapusan data pengguna secara permanen sesuai regulasi privasi.
- send_message.php: Memproses pengiriman pesan dari halaman kontak ke sistem.

Direktori Aset dan Data

- database/ (arvo.sql): Berisi berkas skema database SQL yang digunakan untuk melakukan import struktur tabel dan relasi pada MySQL.
- images/: Menyimpan aset grafis vektor seperti Logoo.svg untuk identitas brand dan Maskot.svg sebagai elemen pendukung visual dan karakter aplikasi.

4.2 Kode Program yang Penting

Bagian ini menjelaskan analisis terhadap file-file program inti pada sistem ARVO – Activity & Goals Manager. Analisis difokuskan pada script backend utama yang menangani proses logika sistem, validasi, keamanan, serta komunikasi data antara frontend dan database.

4.2.1 Modul Autentikasi Pengguna

a) File backend/register.php

```
1 <?php
2 header("Content-Type: application/json");
3 require "../config.php"; //sesuaikan path
4
5 // Ambil data dari form
6 $username = $_POST['regUsername'] ?? '';
7 $email = $_POST['regEmail'] ?? '';
8 $password = $_POST['regPassword'] ?? '';
9 $confirm = $_POST['regConfirmPassword'] ?? '';
10
11 // Cek password & konfirmasi
12 if ($password != $confirm) {
13     echo json_encode(["status" => "error", "message" => "Password tidak cocok!"]);
14     exit;
15 }
16
17 // Cek apakah username sudah ada
18 $stmt = $conn->prepare("SELECT id FROM users WHERE username = ?");
19 $stmt->bind_param("s", $username);
20 $stmt->execute();
21 $stmt->store_result();
22
23 if ($stmt->num_rows > 0) {
24     echo json_encode(["status" => "error", "message" => "Username sudah digunakan!"]);
25     exit;
26 }
27
28 // Cek apakah email sudah ada
29 $stmt = $conn->prepare("SELECT id FROM users WHERE email = ?");
30 $stmt->bind_param("s", $email);
31 $stmt->execute();
32 $stmt->store_result();
33
34 if ($stmt->num_rows > 0) {
35     echo json_encode(["status" => "error", "message" => "Email sudah digunakan!"]);
36     exit;
37 }
38
39 // Hash password
40 $hashedPassword = password_hash($password, PASSWORD_DEFAULT);
41
42 // Simpan ke database
43 $stmt = $conn->prepare("INSERT INTO users (username, email, password_hash) VALUES (?, ?, ?)");
44 $stmt->bind_param("sss", $username, $email, $hashedPassword);
45
46 if ($stmt->execute()) {
47     echo json_encode(["status" => "success", "message" => "Registrasi berhasil!"]);
48 } else {
49     echo json_encode(["status" => "error", "message" => "Gagal mendaftar."]);
50 }
51 ?>
52
```

Fungsi Utama:

Digunakan untuk proses pendaftaran akun pengguna baru. Script ini menerima data dari form registrasi kemudian memvalidasi dan menyimpan akun ke database.

Analisis Kode:

Kode program di atas merupakan skrip backend berbasis PHP yang berfungsi untuk menangani proses registrasi pengguna baru pada aplikasi ARVO melalui mekanisme API berformat JSON. Secara teknis, alur kerja skrip ini dimulai dengan melakukan validasi kesesuaian antara kata sandi dan konfirmasi kata sandi, kemudian dilanjutkan dengan pemeriksaan ketersediaan username serta alamat email di dalam database menggunakan prepared statements untuk mencegah serangan SQL Injection. Keamanan data menjadi prioritas utama dengan diterapkannya fungsi `password_hash()` menggunakan algoritma `PASSWORD_DEFAULT` (Bcrypt) untuk menyamarkan kata sandi sebelum disimpan secara permanen ke dalam tabel `users`. Jika seluruh tahapan validasi dan proses insert data berhasil dilakukan, sistem akan mengembalikan respons sukses dalam format JSON yang dapat langsung diproses oleh frontend untuk memberikan umpan balik kepada pengguna.

b) File backend/login.php

```
1 <?php
2 session_start();
3 header("Content-Type: application/json");
4 require "../config.php";
5
6 // Ambil data dari form
7 $username = $_POST['username'] ?? '';
8 $password = $_POST['password'] ?? '';
9
10 // Cek user berdasarkan username
11 $stmt = $conn->prepare("SELECT * FROM users WHERE username = ?");
12 $stmt->bind_param("s", $username);
13 $stmt->execute();
14 $result = $stmt->get_result();
15
16 if ($result->num_rows == 0) {
17     echo json_encode(["status" => "error", "message" => "Username tidak ditemukan"]);
18     exit;
19 }
20
21 $user = $result->fetch_assoc();
22
23 // Verifikasi password
24 if (!password_verify($password, $user['password_hash'])) {
25     echo json_encode(["status" => "error", "message" => "Password salah"]);
26     exit;
27 }
28
29 // SET SESSION
30 $_SESSION['user_id'] = $user['id'];
31 $_SESSION['username'] = $user['username'];
32
33 // KIRIM DATA LENGKAP KE FRONTEND
34 echo json_encode([
35     "status" => "success",
36     "message" => "Login berhasil",
37     "user" => [
38         "id" => $user['id'],
39         "username" => $user['username'],
40         "email" => $user['email'],
41         "institution" => $user['institution'],
42         "major" => $user['major'],
43         "bio" => $user['bio'],
44         "created_at" => $user['created_at']
45     ]
46 ]);
47 ?>
```

Fungsi Utama:

Menangani proses autentikasi login pengguna.

Analisis Kode:

Kode program di atas merupakan skrip backend PHP yang berfungsi untuk menangani proses autentikasi pengguna pada aplikasi ARVO dengan standar keamanan tinggi. Alur kerja sistem dimulai dengan inisialisasi sesi (`session_start()`) dan pengambilan kredensial melalui metode POST, yang kemudian divalidasi ke basis data menggunakan prepared statements untuk mencegah serangan SQL Injection. Verifikasi keamanan dilakukan secara mendalam menggunakan fungsi `password_verify()` guna mencocokkan input kata sandi dengan hash yang tersimpan di database, sehingga menjaga integritas data kredensial. Setelah autentikasi berhasil, skrip akan menyimpan identitas pengguna ke dalam variabel `$_SESSION` dan mengirimkan respons sukses dalam format JSON yang berisi data profil lengkap pengguna—termasuk informasi institusi dan biografi—untuk digunakan oleh lapisan frontend.

c) File backend/logout.php



```
1 <?php
2 session_start();
3 session_unset(); // Hapus semua session
4 session_destroy(); // Hancurkan session sepenuhnya
5
6 header("Content-Type: application/json");
7
8 echo json_encode([
9     "status" => "success",
10    "message" => "Logout berhasil"
11 ]);
12 ?>
```

Fungsi Utama:

Menghapus session pengguna sehingga akses dibatasi kembali.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend yang berfungsi untuk menangani terminasi sesi pengguna atau proses logout secara aman pada aplikasi ARVO. Alur kerja skrip ini dimulai dengan menginisialisasi sesi yang sedang berjalan melalui `session_start()`, diikuti dengan pembersihan seluruh variabel sesi menggunakan `session_unset()`, dan diakhiri dengan penghancuran data sesi secara permanen di sisi server melalui instruksi `session_destroy()`. Setelah status autentikasi dihapus sepenuhnya untuk menjamin keamanan akun, skrip memberikan umpan balik kepada lapisan frontend dengan mengirimkan respons berformat JSON yang menyatakan bahwa proses penghentian sesi telah berhasil dilakukan.

4.2.2 Modul Manajemen Jadwal

a) File backend/schedule/add_schedule.php



```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require('../config.php');
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $user_id = $_SESSION['user_id'];
12 $title = $_POST['scheduleTitle'] ?? '';
13 $date = $_POST['scheduleDate'] ?? '';
14 $time = $_POST['scheduleTime'] ?? '';
15 $type = $_POST['scheduleType'] ?? '';
16 $priority = $_POST['schedulePriority'] ?? '';
17 $desc = $_POST['scheduleDescription'] ?? '';
18
19 $stmt = $conn->prepare("
20     INSERT INTO schedules (user_id, title, date, time, type, priority, description)
21     VALUES (?, ?, ?, ?, ?, ?, ?)
22 ");
23 $stmt->bind_param("isssss", $user_id, $title, $date, $time, $type, $priority, $desc);
24
25 if ($stmt->execute()) {
26     echo json_encode([
27         "status" => "success",
28         "message" => "Jadwal berhasil ditambahkan",
29         "schedule" => [
30             "id" => $stmt->insert_id,
31             "title" => $title,
32             "date" => $date,
33             "time" => $time,
34             "type" => $type,
35             "priority" => $priority,
36             "description" => $desc
37         ]
38     ]);
39 } else {
40     echo json_encode(["status" => "error", "message" => "Gagal menambahkan jadwal"]);
41 }
42 ?>
43
```

Fungsi Utama:

Menambahkan jadwal baru ke sistem.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend yang berfungsi untuk menangani penambahan data agenda baru ke dalam tabel schedules pada aplikasi ARVO. Alur kerja skrip ini diawali dengan validasi hak akses melalui pemeriksaan user_id pada sesi aktif untuk memastikan hanya pengguna terautentikasi yang dapat menambah jadwal. Secara teknis, sistem mengambil data input dari formulir (seperti judul, tanggal, waktu, tipe, dan prioritas) lalu menyimpannya ke database menggunakan prepared statements guna menjamin keamanan dari serangan SQL Injection. Jika operasi berhasil, skrip akan mengembalikan respons berformat JSON yang berisi detail jadwal yang baru saja disimpan untuk kemudian dirender secara dinamis oleh lapisan frontend.

b) File backend/schedule/get_schedule.php

```
1  <?php
2  header("Content-Type: application/json");
3  session_start();
4  require('../../config.php');
5
6  if (!isset($_SESSION['user_id'])) {
7      echo json_encode([
8          "status" => "error",
9          "message" => "Unauthorized"
10     ]);
11     exit;
12 }
13
14 $user_id = $_SESSION['user_id'];
15
16 // ===== TAMBAHKAN: Handle GET single schedule by ID =====
17 if (isset($_GET['id']) && !empty($_GET['id'])) {
18     $id = intval($_GET['id']);
19
20     $stmt = $conn->prepare("SELECT * FROM schedules WHERE id = ? AND user_id = ?");
21     $stmt->bind_param("ii", $id, $user_id);
22     $stmt->execute();
23     $result = $stmt->get_result();
24
25     if ($result->num_rows > 0) {
26         $schedule = $result->fetch_assoc();
27         echo json_encode([
28             "status" => "success",
29             "schedule" => $schedule,
30             "message" => "Schedule found"
31         ]);
32     } else {
33         echo json_encode([
34             "status" => "error",
35             "message" => "Schedule not found or you don't have permission"
36         ]);
37     }
38     exit;
39 }
40 // ===== END OF SINGLE SCHEDULE HANDLING =====
41
42 // Original code for getting ALL schedules
43 $stmt = $conn->prepare("SELECT * FROM schedules WHERE user_id = ? ORDER BY date, time");
44 $stmt->bind_param("i", $user_id);
45 $stmt->execute();
46 $result = $stmt->get_result();
47
48 $schedules = [];
49 while ($row = $result->fetch_assoc()) {
50     $schedules[] = $row;
51 }
52
53 echo json_encode([
54     "status" => "success",
55     "user" => [
56         "id" => $_SESSION['user_id'],
57         "username" => $_SESSION['username']
58     ],
59     "schedules" => $schedules
60 ]);
61 ?>
```


Fungsi Utama:

Mengambil seluruh data jadwal milik user.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk mengelola pengambilan data jadwal dari basis data dengan tingkat keamanan yang terjamin. Alur kerja sistem diawali dengan validasi sesi untuk memastikan hanya pengguna terautentikasi yang dapat mengakses data, sehingga mencegah akses tidak sah (unauthorized access). Skrip ini memiliki dua fungsionalitas utama: mengambil satu data jadwal spesifik berdasarkan parameter id melalui metode GET, atau mengambil seluruh daftar jadwal milik pengguna yang kemudian diurutkan secara kronologis berdasarkan tanggal dan waktu. Secara teknis, penggunaan prepared statements dengan parameter user_id memastikan bahwa data yang dikembalikan hanya milik pengguna yang bersangkutan, sekaligus memitigasi risiko serangan SQL Injection. Seluruh output dikirimkan dalam format JSON agar dapat diproses secara asinkron oleh antarmuka frontend.

c) File backend/schedule/edit_schedule.php

A screenshot of a code editor showing a PHP script for editing a schedule. The script starts with a header setting 'Content-Type: application/json', session start, and a require statement for a config file. It then checks if the user is logged in (isset(\$_SESSION['user_id'])). If not, it returns an unauthorized JSON response. If logged in, it gets the user_id from the session. It then sanitizes various input fields from \$_POST: id, title, date, time, type, priority, description, and completed. A prepared statement is created to update the 'schedules' table with these values, filtered by the user's id. The statement is executed, and if successful, it returns a success JSON response. If there's an error, it returns an error JSON response with the database error message.

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require('../config.php');
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $user_id = $_SESSION['user_id'];
12
13 $id = $_POST['id'] ?? '';
14 $title = $_POST['title'] ?? '';
15 $date = $_POST['date'] ?? '';
16 $time = $_POST['time'] ?? '';
17 $type = $_POST['type'] ?? '';
18 $priority = $_POST['priority'] ?? '';
19 $desc = $_POST['description'] ?? '';
20 $completed = isset($_POST['completed']) ? 1 : 0;
21
22 $stmt = $conn->prepare("
23     UPDATE schedules
24     SET title=?, date=?, time=?, type=?, priority=?, description=?, completed=?
25     WHERE id=? AND user_id=?
26 ");
27 $stmt->bind_param("ssssssii", $title, $date, $time, $type, $priority, $desc, $completed, $id, $user_id);
28
29 if ($stmt->execute()) {
30     echo json_encode(["status" => "success", "message" => "Jadwal berhasil di-update"]);
31 } else {
32     // TAMPILKAN error detail untuk debugging
33     echo json_encode(["status" => "error", "message" => "Gagal update jadwal: " . $stmt->error]);
34 }
35 ?>
```

Fungsi Utama:

Mengubah jadwal yang sudah ada.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk memperbarui data agenda yang sudah ada di dalam basis data. Alur kerja sistem dimulai dengan memverifikasi identitas pengguna melalui sesi aktif untuk memastikan otoritas akses, kemudian menangkap berbagai parameter input seperti judul, waktu, kategori, hingga status penyelesaian (completed) melalui metode POST. Secara teknis, skrip ini mengimplementasikan prepared statements dalam eksekusi perintah UPDATE untuk menjamin keamanan dari manipulasi kueri atau serangan SQL Injection. Integrasi parameter `user_id` di dalam klausa WHERE memastikan bahwa perubahan data hanya dapat dilakukan pada jadwal milik pengguna yang bersangkutan, sehingga menjaga integritas dan privasi data dalam sistem.

d) File backend/schedule/delete_schedule.php

```
1  <?php
2  header("Content-Type: application/json");
3  session_start();
4  require('../config.php');
5
6  if (!isset($_SESSION['user_id'])) {
7      echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8      exit;
9  }
10
11  $user_id = $_SESSION['user_id'];
12  $id = $_POST['id'] ?? '';
13
14  $stmt = $conn->prepare("DELETE FROM schedules WHERE id=? AND user_id=?");
15  $stmt->bind_param("ii", $id, $user_id);
16
17  if ($stmt->execute()) {
18      echo json_encode(["status" => "success", "message" => "Jadwal berhasil dihapus"]);
19  } else {
20      echo json_encode(["status" => "error", "message" => "Gagal menghapus jadwal"]);
21  }
22  ?>
23
```

Fungsi Utama:

Menghapus jadwal berdasarkan ID.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk menangani penghapusan data agenda secara permanen dari basis data. Alur kerja sistem dimulai dengan memverifikasi autentikasi pengguna melalui sesi aktif guna memastikan otoritas akses, sehingga mencegah tindakan penghapusan oleh pihak yang

tidak berwenang. Secara teknis, skrip ini mengimplementasikan prepared statements dalam eksekusi perintah DELETE untuk menjamin keamanan dari serangan SQL Injection. Penggunaan parameter ganda, yaitu id jadwal dan user_id, dalam klausa WHERE merupakan langkah krusial untuk memastikan bahwa seorang pengguna hanya dapat menghapus data miliknya sendiri, sehingga integritas dan privasi data antar pengguna tetap terjaga.

e) File backedn/schedule/toggle_completed.php



```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require('../config.php');
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $id = $_POST['id'];
12
13 $stmt = $conn->prepare("UPDATE schedules SET completed = NOT completed WHERE id=? AND user_id=?");
14 $stmt->bind_param("ii", $id, $_SESSION['user_id']);
15 $stmt->execute();
16
17 echo json_encode(["status" => "success"]);
```

Fungsi Utama:

Menandai jadwal selesai / belum selesai.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk mengubah status penyelesaian (toggle completion status) sebuah jadwal secara dinamis. Alur kerja sistem dimulai dengan memverifikasi autentikasi pengguna melalui pemeriksaan sesi aktif guna menjamin keamanan dan mencegah akses tidak sah. Secara teknis, kode ini menggunakan perintah UPDATE dengan logika pembalikan nilai Boolean (NOT completed) pada basis data, yang memungkinkan status tugas berubah secara bergantian antara selesai dan belum selesai hanya dengan satu request. Penggunaan prepared statements dengan parameter ganda (id dan user_id) memastikan bahwa perubahan status hanya dapat dilakukan pada data milik pengguna yang bersangkutan, sehingga menjaga integritas data serta melindungi sistem dari serangan SQL Injection.

4.2.3 Modul Notulensi

a) File backend/notes/add_note.php

A screenshot of a code editor with a dark background and light-colored text. The code is written in PHP and is numbered from 1 to 38. It starts with a header declaration for JSON, session start, and a require statement for a config file. It then checks if the user is logged in by checking the session. If not, it returns an error response. If logged in, it takes input from the POST method for title, category, and content. It then uses a prepared statement to insert this data into a table named 'notes'. Finally, it checks if the insertion was successful and returns a success or error response in JSON format.

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require("../config.php");
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode([
8         "status" => "error",
9         "message" => "Unauthorized"
10    ]);
11    exit;
12 }
13
14 $user_id = $_SESSION['user_id'];
15 $title = $_POST['noteTitle'] ?? '';
16 $category = $_POST['noteCategory'] ?? '';
17 $content = $_POST['noteContent'] ?? '';
18
19 $stmt = $conn->prepare("
20     INSERT INTO notes (user_id, title, category, content)
21     VALUES (?, ?, ?, ?)
22 ");
23
24 $stmt->bind_param("isss", $user_id, $title, $category, $content);
25
26 if ($stmt->execute()) {
27     echo json_encode([
28         "status" => "success",
29         "message" => "Notulensi berhasil ditambahkan"
30    ]);
31 } else {
32     echo json_encode([
33         "status" => "error",
34         "message" => "Gagal menambahkan notulensi"
35    ]);
36 }
37 >>
38
```

Fungsi Utama:

Menambahkan catatan baru seperti catatan rapat, kuliah, dll.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk menangani proses penambahan data notulensi baru ke dalam tabel notes di basis data. Alur kerja sistem dimulai dengan memverifikasi status autentikasi pengguna melalui pemeriksaan sesi (session_start) guna memastikan keamanan data dan mencegah akses tanpa izin. Secara teknis, kode ini menangkap input dari formulir berupa judul, kategori, dan konten catatan, lalu mengeksekusi perintah INSERT menggunakan prepared statements untuk menjamin perlindungan terhadap serangan SQL Injection. Keberhasilan operasi akan menghasilkan respons berformat JSON yang dapat diproses oleh lapisan frontend untuk memberikan umpan balik instan kepada pengguna.

b) File backend/notes/get_note.php

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require('.../config.php');
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $user_id = $_SESSION['user_id'];
12
13 // ===== TAMBAHAKAN: Handle GET single note by ID =====
14 if (isset($_GET['id']) && !empty($_GET['id'])) {
15     $id = intval($_GET['id']);
16
17     $stmt = $conn->prepare("
18         SELECT id, title, category, content, created_at
19         FROM notes
20         WHERE id = ? AND user_id = ?
21     ");
22     $stmt->bind_param("ii", $id, $user_id);
23     $stmt->execute();
24     $result = $stmt->get_result();
25
26     if ($result->num_rows > 0) {
27         $note = $result->fetch_assoc();
28         echo json_encode([
29             "status" => "success",
30             "note" => $note,
31             "message" => "Note found"
32         ]);
33     } else {
34         echo json_encode([
35             "status" => "error",
36             "message" => "Note not found or you don't have permission"
37         ]);
38     }
39     exit;
40 }
41 // ===== END OF SINGLE NOTE HANDLING =====
42
43 // Original code for getting ALL notes
44 $stmt = $conn->prepare("
45     SELECT id, title, category, content, created_at
46     FROM notes
47     WHERE user_id = ?
48     ORDER BY created_at DESC
49 ");
50 $stmt->bind_param("i", $user_id);
51 $stmt->execute();
52 $result = $stmt->get_result();
53
54 $notes = [];
55 while ($row = $result->fetch_assoc()) {
56     $notes[] = $row;
57 }
58
59 echo json_encode([
60     "status" => "success",
61     "notes" => $notes
62 ]);
63 }
```

Fungsi Utama:

Mengambil seluruh data notulensi user.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk mengelola pengambilan data notulensi dari basis data dengan tingkat keamanan yang terjamin. Alur kerja sistem diawali dengan verifikasi sesi aktif untuk memastikan bahwa hanya pengguna terautentikasi yang dapat mengakses informasi, sehingga mencegah akses data secara ilegal. Modul ini memiliki fungsionalitas ganda: dapat mengambil satu catatan spesifik berdasarkan parameter id melalui metode GET, atau mengambil seluruh daftar catatan milik pengguna yang diurutkan secara kronologis terbalik berdasarkan waktu pembuatan. Secara teknis, implementasi prepared statements dengan validasi parameter `user_id` memastikan integritas data tetap terjaga dan melindungi sistem dari serangan SQL Injection, sementara seluruh output dikirimkan dalam format JSON untuk diproses secara dinamis oleh antarmuka frontend.

c) File backend/notes/edit_note.php

```
1 <?php
2 session_start();
3 header("Content-Type: application/json; charset=UTF-8");
4
5 // Nonaktifkan error reporting untuk production
6 // error_reporting(0);
7
8 // Cek session
9 if (!isset($_SESSION['user_id'])) {
10     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
11     exit;
12 }
13
14 require("../config.php");
15
16 $user_id = $_SESSION['user_id'];
17
18 // Ambil data langsung dari $_POST
19 $id = $_POST['id'] ?? '';
20 $title = $_POST['title'] ?? '';
21 $category = $_POST['category'] ?? '';
22 $content = $_POST['content'] ?? '';
23
24 // Validasi sederhana
25 if (empty($id) || empty($title) || empty($category) || empty($content)) {
26     echo json_encode(["status" => "error", "message" => "Semua field wajib diisi"]);
27     exit;
28 }
29
30 $id = intval($id);
31
32 // Update database
33 $stmt = $conn->prepare("UPDATE notes SET title=?, category=?, content=? WHERE id=? AND user_id=?");
34 $stmt->bind_param("sssi", $title, $category, $content, $id, $user_id);
35
36 if ($stmt->execute()) {
37     echo json_encode(["status" => "success", "message" => "Notulensi berhasil diupdate"]);
38 } else {
39     echo json_encode(["status" => "error", "message" => "Gagal: " . $stmt->error]);
40 }
41
42 $stmt->close();
43 exit;
44 ?>
```

Fungsi Utama:

Mengubah isi catatan.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk memperbarui data catatan yang sudah ada di dalam basis data. Alur kerja sistem diawali dengan validasi keamanan melalui pemeriksaan sesi aktif (session_start) guna memastikan bahwa hanya pengguna terautentikasi yang memiliki hak akses (authorization). Secara teknis, skrip ini melakukan validasi input untuk memastikan seluruh field wajib terisi sebelum mengeksekusi perintah UPDATE menggunakan prepared statements. Penggunaan parameter ganda (id dan user_id) dalam kueri SQL menjamin integritas data dengan memastikan pengguna hanya dapat mengubah catatan miliknya sendiri, sekaligus memberikan perlindungan maksimal terhadap serangan SQL Injection. Hasil eksekusi kemudian dikembalikan dalam format JSON untuk diproses lebih lanjut oleh antarmuka frontend. modul

d) File backedn/notes//delete_note.php

A screenshot of a code editor showing a PHP script for deleting a note. The script starts with a header for JSON output, session initialization, and a configuration file include. It checks if the user is logged in; if not, it returns an 'Unauthorized' error. If logged in, it takes the note ID from the POST request and prepares a SQL statement to delete the note from the 'notes' table, ensuring the user is the owner. The statement is executed, and a success or error message is returned in JSON format.

```
1  <?php
2  header("Content-Type: application/json");
3  session_start();
4  require('../config.php');
5
6  if (!isset($_SESSION['user_id'])) {
7      echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8      exit;
9  }
10
11  $user_id = $_SESSION['user_id'];
12  $id = $_POST['id'] ?? '';
13
14  $stmt = $conn->prepare("
15      DELETE FROM notes
16      WHERE id=? AND user_id=?
17  ");
18
19  $stmt->bind_param("ii", $id, $user_id);
20
21  if ($stmt->execute()) {
22      echo json_encode(["status" => "success", "message" => "Notulensi berhasil dihapus"]);
23  } else {
24      echo json_encode(["status" => "error", "message" => "Gagal menghapus notulensi"]);
25  }
26  ?>
27
```

Fungsi Utama:

Menghapus catatan.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk menangani penghapusan data catatan secara permanen dari tabel notes dalam basis data. Alur kerja sistem dimulai dengan memverifikasi autentikasi pengguna melalui pemeriksaan sesi aktif guna memastikan otoritas akses dan mencegah tindakan penghapusan oleh pihak yang tidak berwenang. Secara teknis, skrip ini mengimplementasikan prepared statements dalam eksekusi perintah DELETE untuk menjamin keamanan dari serangan SQL Injection. Penggunaan parameter ganda, yaitu id catatan dan user_id, dalam klausa WHERE merupakan langkah krusial untuk memastikan bahwa seorang pengguna hanya dapat menghapus data miliknya sendiri, sehingga integritas dan privasi data antar pengguna tetap terjaga di dalam sistem. modul

4.2.4 Modul Target

- a) File backend/goals/add_goal.php

A screenshot of a code editor with a dark background and light-colored text. The code is a PHP script for adding a goal to a database. It starts with a header, session start, and a check for user authentication. If not authenticated, it returns an error. If authenticated, it takes input from the POST method, sanitizes it, and inserts it into the 'goals' table using a prepared statement. It then returns a success message or an error message if the insertion fails.

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require("../config.php");
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $user_id = $_SESSION['user_id'];
12 $type = $_POST['goalType'] ?? ''; // harus sesuai enum: semester|academic|career|personal
13 $title = $_POST['goalTitle'] ?? '';
14 $description = $_POST['goalDescription'] ?? ''; // atau goalNotes, tapi disarankan
15 $deadline = $_POST['goalDeadline'] ?? null;
16 $progress = 0;
17
18 $stmt = $conn->prepare("
19     INSERT INTO goals (user_id, type, title, description, deadline, progress)
20     VALUES (?, ?, ?, ?, ?, ?)
21 ");
22 $stmt->bind_param("issssi", $user_id, $type, $title, $description, $deadline, $progress);
23
24 if ($stmt->execute()) {
25     echo json_encode(["status" => "success", "message" => "Target berhasil ditambahkan"]);
26 } else {
27     echo json_encode(["status" => "error", "message" => "Gagal menambahkan target"]);
28 }
29 ?>
```

Fungsi Utama:

Membuat target baru beserta progress awal.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk menangani proses penambahan data target baru ke dalam tabel goals di basis data. Alur kerja sistem dimulai dengan memverifikasi status autentikasi melalui pemeriksaan

sesi aktif (user_id) guna memastikan bahwa hanya pengguna resmi yang dapat menambah data. Nilai progres diinisialisasi secara otomatis pada angka nol, dan hasil akhir dari operasi tersebut dikembalikan dalam format JSON untuk memberikan umpan balik langsung pada antarmuka frontend.

b) File backend/goals/get_goal.php

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require('../config.php');
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $user_id = $_SESSION['user_id'];
12
13 // ===== [AMBAHAN: Handle GET single goal by ID] =====
14 if (isset($_GET['id']) && !empty($_GET['id'])) {
15     $id = intval($_GET['id']);
16
17     $stmt = $conn->prepare("SELECT * FROM goals WHERE id = ? AND user_id = ?");
18     $stmt->bind_param("ii", $id, $user_id);
19     $stmt->execute();
20     $result = $stmt->get_result();
21
22     if ($result->num_rows > 0) {
23         $goal = $result->fetch_assoc();
24         echo json_encode([
25             "status" => "success",
26             "goal" => $goal,
27             "message" => "Goal found"
28         ]);
29     } else {
30         echo json_encode([
31             "status" => "error",
32             "message" => "Goal not found or you don't have permission"
33         ]);
34     }
35     exit;
36 }
37 // ===== END OF SINGLE GOAL HANDLING =====
38
39 // Original code for getting All goals (sesuaikan dengan kode Anda)
40 $stmt = $conn->prepare("SELECT * FROM goals WHERE user_id = ? ORDER BY created_at DESC");
41 $stmt->bind_param("i", $user_id);
42 $stmt->execute();
43 $result = $stmt->get_result();
44
45 $goals = [];
46 while ($row = $result->fetch_assoc()) {
47     $goals[] = $row;
48 }
49
50 echo json_encode([
51     "status" => "success",
52     "goals" => $goals
53 ]);
54 }
```

Fungsi Utama:

Mengambil semua target milik user.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk mengelola pengambilan data target (goals) dari basis data dengan menerapkan

protokol keamanan berbasis sesi. Alur kerja sistem dimulai dengan verifikasi identitas pengguna melalui session_start() untuk memastikan hanya pemilik data yang sah yang dapat mengakses informasi tersebut, sehingga mencegah potensi kebocoran data (unauthorized access). Secara teknis, skrip ini mendukung fungsionalitas ganda: mengambil satu data target spesifik berdasarkan parameter id melalui metode GET, atau mengambil seluruh daftar target milik pengguna yang diurutkan berdasarkan waktu pembuatan terbaru. Penggunaan prepared statements dengan validasi ganda pada kolom id dan user_id menjamin bahwa kueri aman dari serangan SQL Injection sekaligus memastikan integritas data tetap terjaga, sebelum akhirnya dikirimkan kembali ke antarmuka frontend dalam format JSON.

c) File backend/goals/edit_goal.php

A screenshot of a code editor showing a PHP script for editing goals. The script starts with a header setting the content type to application/json, followed by session_start() and a require statement for config.php. It then checks if the user is logged in (isset(\$_SESSION['user_id'])). If not, it returns a JSON error response. If logged in, it retrieves the user ID and sanitizes POST data for id, type, title, description, deadline, and progress. It then prepares an SQL UPDATE statement for the 'goals' table, using prepared statements to prevent SQL injection. The statement is executed, and a success or error JSON response is returned based on the result.

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require("../config.php");
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $user_id = $_SESSION['user_id'];
12 $id = $_POST['id'] ?? 0;
13 $type = $_POST['type'] ?? '';
14 $title = $_POST['title'] ?? '';
15 $description = $_POST['description'] ?? '';
16 $deadline = $_POST['deadline'] ?? null;
17 $progress = intval($_POST['progress'] ?? 0);
18
19 // Jika deadline kosong, set ke null
20 if (empty($deadline)) {
21     $deadline = null;
22 }
23
24 $stmt = $conn->prepare("
25     UPDATE goals
26     SET type=?, title=?, description=?, deadline=?, progress=?
27     WHERE id=? AND user_id=?
28 ");
29
30 $stmt->bind_param("ssssiii", $type, $title, $description, $deadline, $progress, $id, $user_id);
31
32 if ($stmt->execute()) {
33     echo json_encode(["status" => "success", "message" => "Target berhasil diperharui"]);
34 } else {
35     echo json_encode(["status" => "error", "message" => "Gagal update target: " . $stmt->error]);
36 }
37 ?>
```

Fungsi Utama:

Mengubah progress dan data target.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk memperbarui data target (goals) yang telah ada di dalam basis data. Alur kerja sistem dimulai dengan verifikasi keamanan melalui pemeriksaan sesi aktif guna memastikan otoritas akses pengguna sebelum memproses berbagai parameter input

seperti tipe, judul, deskripsi, tenggat waktu, dan persentase progres. Secara teknis, skrip ini menggunakan prepared statements untuk mengeksekusi perintah UPDATE dengan validasi ganda pada parameter id dan user_id, sehingga menjamin bahwa pengguna hanya dapat memodifikasi data miliknya sendiri sekaligus melindungi sistem dari serangan SQL Injection. Hasil dari operasi pembaruan ini dikembalikan dalam format JSON untuk memberikan konfirmasi status secara real-time kepada antarmuka frontend.

d) File backedn/goals/delete_ goal.php

A screenshot of a code editor showing a PHP script for deleting a goal. The script starts with a PHP tag, sets the content type to application/json, starts a session, and requires a config file. It checks if the user is logged in (has a session user_id). If not, it returns an unauthorized JSON response. If logged in, it gets the user_id from the session and the goal_id from the POST request. It then prepares a DELETE statement for the 'goals' table, binding the goal_id and user_id as parameters. It executes the statement and returns a JSON response indicating success or failure. The script ends with a closing PHP tag.

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require('../config.php');
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $user_id = $_SESSION['user_id'];
12 $id = $_POST['id'] ?? 0;
13
14 $stmt = $conn->prepare("DELETE FROM goals WHERE id = ? AND user_id = ?");
15 $stmt->bind_param("ii", $id, $user_id);
16
17 if ($stmt->execute()) {
18     echo json_encode(["status" => "success", "message" => "Target berhasil dihapus"]);
19 } else {
20     echo json_encode(["status" => "error", "message" => "Gagal menghapus target"]);
21 }
22 ?>
23
```

Fungsi Utama:

Menghapus target.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk menangani penghapusan data target secara permanen dari tabel goals di basis data. Alur kerja sistem dimulai dengan memverifikasi autentikasi pengguna melalui sesi aktif guna memastikan otoritas akses dan mencegah penghapusan data oleh pengguna yang tidak berwenang. Secara teknis, skrip ini menggunakan prepared statements dengan perintah DELETE yang melibatkan parameter ganda, yakni id target dan user_id, untuk menjamin integritas data serta memberikan perlindungan maksimal terhadap serangan SQL Injection. Setelah proses eksekusi di server selesai, sistem akan mengembalikan respons dalam format JSON yang mengonfirmasi keberhasilan atau kegagalan operasi kepada antarmuka frontend.

4.2.5 Modul Dashboard & Statistik

File backend/get_stats.php

```
1 <?php
2 header('Content-Type: application/json');
3 session_start();
4 require "../config.php";
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $user = [
12     "id" => $_SESSION['user_id'],
13     "username" => $_SESSION['username']
14 ];
15
16 $user_id = (int)$_SESSION['user_id'];
17
18 // =====
19 // Query statistics
20 // =====
21
22 $stmt = $conn->prepare("SELECT COUNT(*) AS total FROM schedules WHERE user_id = ?");
23 $stmt->bind_param("i", $user_id);
24 $stmt->execute();
25 $totalSchedules = $stmt->get_result()->fetch_assoc()['total']; // 0;
26
27 $stmt = $conn->prepare("SELECT COUNT(*) AS total FROM notes WHERE user_id = ?");
28 $stmt->bind_param("i", $user_id);
29 $stmt->execute();
30 $totalNotes = $stmt->get_result()->fetch_assoc()['total']; // 0;
31
32 $stmt = $conn->prepare("SELECT COUNT(*) AS total FROM goals WHERE user_id = ?");
33 $stmt->bind_param("i", $user_id);
34 $stmt->execute();
35 $totalGoals = $stmt->get_result()->fetch_assoc()['total']; // 0;
36
37 $stmt = $conn->prepare("SELECT COUNT(*) AS total FROM goals WHERE user_id = ? AND progress = 100");
38 $stmt->bind_param("i", $user_id);
39 $stmt->execute();
40 $completedGoals = $stmt->get_result()->fetch_assoc()['total']; // 0;
41
42 $stmt = $conn->prepare("SELECT AVG(progress) AS avgProgress FROM goals WHERE user_id = ?");
43 $stmt->bind_param("i", $user_id);
44 $stmt->execute();
45 $averageGoalProgress = round($stmt->get_result()->fetch_assoc()['avgProgress'], 2); // 0;
46
47 $stmt = $conn->prepare("SELECT COUNT(*) AS high FROM schedules WHERE user_id = ? AND priority = 'high'");
48 $stmt->bind_param("i", $user_id);
49 $stmt->execute();
50 $highPrioritySchedules = $stmt->get_result()->fetch_assoc()['high']; // 0;
51
52 $stmt = $conn->prepare("SELECT COUNT(*) AS days FROM user_activity WHERE user_id = ?");
53 $stmt->bind_param("i", $user_id);
54 $stmt->execute();
55 $activeDays = $stmt->get_result()->fetch_assoc()['days']; // 0;
56
57 $stmt = $conn->prepare("SELECT COUNT(*) AS completed FROM schedules WHERE user_id = ? AND completed = 1");
58 $stmt->bind_param("i", $user_id);
59 $stmt->execute();
60 $completedTasks = $stmt->get_result()->fetch_assoc()['completed']; // 0;
61
62 echo json_encode([
63     "status" => "success",
64     "user" => $user,
65     "stats" => [
66         "totalSchedules" => $totalSchedules,
67         "totalNotes" => $totalNotes,
68         "totalGoals" => $totalGoals,
69         "completedGoals" => $completedGoals,
70         "highPrioritySchedules" => $highPrioritySchedules,
71         "activeDays" => $activeDays,
72         "averageGoalProgress" => $averageGoalProgress,
73         "completedTasks" => $completedTasks
74     ]
75 ]);
76 ?>
```

Fungsi Utama:

Menghasilkan data statistik untuk dashboard.

Analisis Kode:

Skrip PHP tersebut berfungsi sebagai modul backend untuk menyediakan data statistik ringkasan aktivitas pengguna pada aplikasi ARVO melalui API berformat JSON. Alur kerja

dimulai dengan validasi autentikasi berbasis sesi untuk memastikan keamanan akses, kemudian dilanjutkan dengan serangkaian kueri basis data menggunakan prepared statements untuk menghitung akumulasi data dari berbagai tabel, seperti jumlah jadwal (schedules), total notulensi (notes), progres target (goals), hingga tingkat keaktifan pengguna. Secara teknis, skrip ini mengagregasi parameter spesifik seperti rata-rata progres target dan jumlah tugas prioritas tinggi berdasarkan identitas pengguna (user_id) yang sedang masuk, guna menyajikan ringkasan performa yang dipersonalisasi untuk ditampilkan pada komponen dashboard di lapisan frontend.

4.2.6 Modul Profil Pengguna

- a) File backend/goals/get_profile.php



```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require("../config.php");
5
6 if (!isset($_SESSION['user_id'])) {
7     http_response_code(401);
8     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
9     exit;
10 }
11
12 $user_id = $_SESSION['user_id'];
13
14 $stmt = $conn->prepare("SELECT id, username, email, institution, major, bio, created_at FROM users WHERE id = ?");
15 $stmt->bind_param("i", $user_id);
16 $stmt->execute();
17 $rows = $stmt->get_result();
18
19 if ($res = $user = $res->fetch_assoc()) {
20     echo json_encode(["status" => "success", "user" => $user]);
21 } else {
22     echo json_encode(["status" => "error", "message" => "User not found"]);
23 }
24 ?>
```

Fungsi Utama:

Mengambil data profil user.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk mengambil data profil lengkap pengguna yang sedang aktif melalui API berformat JSON. Alur kerja dimulai dengan memverifikasi autentikasi pengguna menggunakan sesi aktif, di mana kegagalan validasi akan mengembalikan kode status HTTP 401 (Unauthorized) untuk menjamin keamanan akses data pribadi. Secara teknis, sistem mengeksekusi prepared statement untuk mengambil kolom spesifik seperti username, email, institusi, jurusan, dan biografi dari tabel users berdasarkan identitas pengguna (user_id) yang tersimpan dalam sesi. Penggunaan metode ini memastikan bahwa data yang disajikan akurat dan terlindungi dari serangan SQL Injection, serta memungkinkan antarmuka frontend untuk menampilkan informasi profil pengguna secara dinamis.

b) File backend/goals/update_profile.php

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require('../config.php');
5
6 if (!isset($_SESSION['user_id'])) {
7     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
8     exit;
9 }
10
11 $user_id = $_SESSION['user_id'];
12
13 $username = trim($_POST['profileName']) ?? '';
14 $email = trim($_POST['profileEmail']) ?? '';
15 $institution = trim($_POST['profileInstitution']) ?? '';
16 $major = trim($_POST['profileMajor']) ?? '';
17 $bio = trim($_POST['profileBio']) ?? '';
18
19 if ($username == '' || $email == '') {
20     echo json_encode(["status" => "error", "message" => "Nama dan email wajib diisi"]);
21     exit;
22 }
23
24 if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
25     echo json_encode(["status" => "error", "message" => "Email tidak valid"]);
26     exit;
27 }
28
29 // Cek email sudah dipakai atau belum
30 $stmt = $conn->prepare("SELECT id FROM users WHERE email = ? AND id <> ?");
31 $stmt->bind_param("si", $email, $user_id);
32 $stmt->execute();
33 $stmt->store_result();
34
35 if ($stmt->num_rows > 0) {
36     echo json_encode(["status" => "error", "message" => "Email sudah digunakan"]);
37     exit;
38 }
39
40 // UPDATE
41 $stmt = $conn->prepare("
42     UPDATE users
43     SET username=?, email=?, institution=?, major=?, bio=?
44     WHERE id=?
45 ");
46 $stmt->bind_param("sssssi", $username, $email, $institution, $major, $bio, $user_id);
47
48 if ($stmt->execute()) {
49     echo json_encode(["status" => "success", "message" => "Profil berhasil diperbarui"]);
50 } else {
51     echo json_encode(["status" => "error", "message" => "Gagal update profil"]);
52 }
53 ?>
```

Fungsi Utama:

Mengubah biodata pengguna.

Analisis Kode:

Skrip PHP tersebut merupakan modul backend pada aplikasi ARVO yang berfungsi untuk mengelola pembaruan data profil pengguna secara aman. Alur kerja sistem dimulai dengan verifikasi sesi aktif dan pengambilan data melalui metode POST, yang kemudian divalidasi secara ketat mencakup pengecekan field kosong, format email menggunakan FILTER_VALIDATE_EMAIL, serta pemeriksaan keunikan email di database guna menghindari duplikasi akun. Secara teknis, skrip ini menggunakan prepared statements untuk mengeksekusi perintah UPDATE pada tabel users berdasarkan identitas pengguna (user_id), sehingga menjamin keamanan dari serangan SQL Injection sekaligus memastikan bahwa pengguna hanya dapat memodifikasi informasi milik mereka sendiri sebelum mengembalikan respons sukses atau gagal dalam format JSON.

c) File backend/goals/change_password.php

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4 require('.../config.php');
5
6 if (!isset($_SESSION['user_id'])) {
7     http_response_code(401);
8     echo json_encode(["status" => "error", "message" => "Unauthorized"]);
9     exit;
10 }
11
12 $user_id = (int) $_SESSION['user_id'];
13
14 $current = $_POST['currentPassword'] ?? '';
15 $new = $_POST['newPassword'] ?? '';
16 $confirm = $_POST['confirmPassword'] ?? '';
17
18 if (!$current || !$new || !$confirm) {
19     http_response_code(400);
20     echo json_encode(["status" => "error", "message" => "Semua field password wajib diisi"]);
21     exit;
22 }
23
24 if ($new != $confirm) {
25     http_response_code(400);
26     echo json_encode(["status" => "error", "message" => "Konfirmasi password tidak cocok"]);
27     exit;
28 }
29
30 if (strlen($new) < 8) {
31     http_response_code(400);
32     echo json_encode(["status" => "error", "message" => "Password baru minimal 8 karakter"]);
33     exit;
34 }
35
36 // Ambil hash password saat ini
37 $stmt = $conn->prepare("SELECT password_hash FROM users WHERE id = ?");
38 $stmt->bind_param("i", $user_id);
39 $stmt->execute();
40 $res = $stmt->get_result();
41 if (!$res || $res->num_rows == 0) {
42     http_response_code(404);
43     echo json_encode(["status" => "error", "message" => "User tidak ditemukan"]);
44     exit;
45 }
46 $row = $res->fetch_assoc();
47 $hash = $row['password_hash'] ?? '';
48
49 // Verifikasi current password
50 if (!password_verify($current, $hash)) {
51     http_response_code(400);
52     echo json_encode(["status" => "error", "message" => "Password saat ini salah"]);
53     exit;
54 }
55
56 // Hash dan update
57 $newHash = password_hash($new, PASSWORD_DEFAULT);
58 $stmt = $conn->prepare("UPDATE users SET password_hash = ? WHERE id = ?");
59 $stmt->bind_param("si", $newHash, $user_id);
60
61 if ($stmt->execute()) {
62     echo json_encode(["status" => "success", "message" => "Password berhasil diubah"]);
63 } else {
64     http_response_code(500);
65     echo json_encode(["status" => "error", "message" => "Gagal mengubah password"]);
66 }
67
```

Fungsi Utama:

Mengubah password dengan validasi: cek password lama, hash password baru, update database.

Analisis Kode:

Skrip PHP tersebut merupakan modul keamanan pada aplikasi ARVO yang berfungsi untuk mengelola proses perubahan kata sandi pengguna secara terproteksi. Alur kerja sistem dimulai dengan validasi sesi dan verifikasi input, mencakup pemeriksaan kelengkapan data, kesesuaian konfirmasi kata sandi, serta pemenuhan standar panjang minimal karakter untuk meningkatkan keamanan akun. Secara teknis, skrip ini

mengimplementasikan logika verifikasi ganda: pertama, mencocokkan kata sandi lama menggunakan fungsi `password_verify()` terhadap hash di database, dan kedua, melakukan enkripsi pada kata sandi baru menggunakan algoritma `password_hash()` sebelum memperbarui tabel users. Penggunaan prepared statements dan kode status HTTP yang spesifik (seperti 401, 403, dan 400) memastikan bahwa proses pembaruan kredensial berjalan aman, kredibel, dan memberikan respons yang tepat kepada lapisan frontend.

4.2.7 Modul Aktivitas

File backend/activity/add_activity.php

```
1 <?php
2 header("Content-Type: application/json");
3 session_start();
4
5 // Debug: Cek session
6 error_log("DEBUG add_activity.php: Session user_id = " . ($SESSION['user_id'] ?? 'NOT SET'));
7
8 if (!isset($SESSION['user_id'])) {
9     echo json_encode(["status" => "error", "message" => "Unauthorized - No session"]);
10    exit;
11 }
12
13 require '../config.php';
14 $user_id = (int)$SESSION['user_id'];
15 $today = date("Y-m-d");
16
17 error_log("DEBUG: user_id=$user_id, today=$today");
18
19 try {
20     // Insert IF NOT EXISTS
21     $stmt = $conn->prepare("
22         INSERT IGNORE INTO user_activity (user_id, activity_date)
23         VALUES (?, ?)
24     ");
25
26     if (!$stmt) {
27         throw new Exception("Prepare failed: " . $conn->error);
28     }
29
30     $stmt->bind_param("is", $user_id, $today);
31
32     if ($stmt->execute()) {
33         $affected_rows = $stmt->affected_rows;
34         error_log("DEBUG: Inserted/ignored. Affected rows: $affected_rows");
35         echo json_encode([
36             "status" => "success",
37             "message" => "Activity logged",
38             "affected_rows" => $affected_rows
39         ]);
40     } else {
41         throw new Exception("Execute failed: " . $stmt->error);
42     }
43
44     $stmt->close();
45 } catch (Exception $e) {
46     error_log("ERROR add_activity.php: " . $e->getMessage());
47     echo json_encode([
48         "status" => "error",
49         "message" => "Database error: " . $e->getMessage()
50     ]);
51 }
52
53 $conn->close();
54 ?>
```


Fungsi Utama:

Mencatat riwayat aktivitas pengguna (digunakan untuk statistik Active Days).

Analisis Kode:

Skrip PHP tersebut merupakan modul pendukung pada aplikasi ARVO yang berfungsi untuk mencatat log aktivitas harian pengguna ke dalam tabel `user_activity` guna keperluan pemantauan keaktifan atau fitur streak. Alur kerja sistem dimulai dengan validasi sesi pengguna dan pengambilan tanggal hari ini, diikuti dengan mekanisme keamanan INSERT IGNORE melalui prepared statements untuk memastikan bahwa data aktivitas hanya dicatat satu kali per hari bagi setiap pengguna tanpa menyebabkan duplikasi atau galat pada basis data. Secara teknis, skrip ini dilengkapi dengan fitur debugging melalui `error_log` dan manajemen eksepsi (try-catch) untuk menangani potensi kegagalan koneksi atau kueri, lalu mengembalikan respons JSON yang mengonfirmasi apakah log baru berhasil ditambahkan atau diabaikan karena sudah ada.

4.2.8 Modul Export & Akun

a) File backend/export_data.php

```
1 <?php
2 require "../config.php";
3
4 $user_id = $_GET['user_id'] ?? 0;
5
6 $data = [];
7
8 // USERS
9 $q = $conn->prepare("SELECT id, username, email, institution, major, bio FROM users WHERE id = ?");
10 $q->bind_param("i", $user_id);
11 $q->execute();
12 $data['user'] = $q->get_result()->fetch_assoc();
13
14 // SCHEDULES
15 $q = $conn->prepare("SELECT * FROM schedules WHERE user_id = ?");
16 $q->bind_param("i", $user_id);
17 $q->execute();
18 $data['schedules'] = $q->get_result()->fetch_all(MYSQLI_ASSOC);
19
20 // NOTES
21 $q = $conn->prepare("SELECT * FROM notes WHERE user_id = ?");
22 $q->bind_param("i", $user_id);
23 $q->execute();
24 $data['notes'] = $q->get_result()->fetch_all(MYSQLI_ASSOC);
25
26 // GOALS
27 $q = $conn->prepare("SELECT * FROM goals WHERE user_id = ?");
28 $q->bind_param("i", $user_id);
29 $q->execute();
30 $data['goals'] = $q->get_result()->fetch_all(MYSQLI_ASSOC);
31
32 header('Content-Type: application/json');
33 header('Content-Disposition: attachment; filename="arvo-export-'. $user_id .'.json"');
34
35 echo json_encode($data, JSON_PRETTY_PRINT);
36 ?>
37
```

Fungsi Utama:

Menjadi fitur tambahan untuk kebutuhan dokumentasi / backup.

Analisis Kode:

Skrip PHP tersebut merupakan modul ekspor data pada aplikasi ARVO yang berfungsi untuk mengumpulkan seluruh informasi milik pengguna dari berbagai tabel dalam basis data ke dalam satu berkas terpusat. Secara teknis, skrip ini menggunakan prepared statements untuk mengambil data profil pengguna, daftar jadwal, kumpulan notulensi, dan target (goals) secara berurutan berdasarkan parameter `user_id` yang diterima melalui metode GET. Hasil agregasi data tersebut kemudian dikonversi menjadi format JSON dan dikirimkan kepada pengguna sebagai berkas unduhan otomatis dengan header `Content-Disposition`, yang memudahkan pengguna untuk mencadangkan atau memigrasikan data pribadi mereka secara terstruktur.

b) File backend/delete_account.php



```
1 <?php
2 session_start();
3 require "../config.php";
4
5 $user_id = $_POST['user_id'] ?? 0;
6
7 if (!$user_id) {
8     echo json_encode(["status" => "error", "message" => "User ID tidak valid"]);
9     exit;
10 }
11
12 $q = $conn->prepare("DELETE FROM users WHERE id = ?");
13 $q->bind_param("i", $user_id);
14
15 if ($q->execute()) {
16     session_destroy();
17     echo json_encode(["status" => "success", "message" => "Akun berhasil dihapus"]);
18 } else {
19     echo json_encode(["status" => "error", "message" => "Gagal menghapus akun"]);
20 }
21 ?>
```

Fungsi Utama:

Menghapus akun beserta semua data yang terkait.

Analisis Kode:

Skrip PHP tersebut merupakan modul terminasi akun pada aplikasi ARVO yang berfungsi untuk menghapus data pengguna secara permanen dari basis data serta mengakhiri sesi yang sedang berjalan. Secara teknis, skrip ini menggunakan prepared

statements untuk mengeksekusi perintah DELETE pada tabel users berdasarkan parameter user_id yang diterima, guna memastikan keamanan dari manipulasi kueri atau serangan SQL Injection. Apabila proses penghapusan di database berhasil, sistem secara otomatis menjalankan fungsi session_destroy() untuk membersihkan seluruh data sesi pengguna pada sisi server, kemudian mengirimkan respons akhir berformat JSON yang mengonfirmasi bahwa akun dan akses pengguna telah berhasil dihapus sepenuhnya.

4.2.9 Frontend Utama

Frontend merupakan komponen yang berfungsi sebagai antarmuka interaksi langsung antara pengguna dengan sistem ARVO. Seluruh proses visualisasi data, input form, navigasi halaman, hingga eksekusi perintah CRUD (Create, Read, Update, Delete) pada jadwal, notulensi, dan goals dilakukan melalui sisi frontend. Frontend sistem ARVO dibangun menggunakan HTML, CSS, dan JavaScript murni (Vanilla JS) tanpa framework, namun tetap menerapkan struktur modular serta pemisahan tanggung jawab kode yang jelas.

1. Struktur Halaman Utama (Index.html)

Index.html berfungsi sebagai basis struktur tunggal aplikasi yang menampung seluruh elemen antarmuka. Dengan total 917 baris kode, file ini menggunakan sistem container-switching untuk menampilkan atau menyembunyikan modul tertentu sesuai status autentikasi pengguna, serta mencakup struktur untuk login modal, navbar, dashboard, dan semua halaman aplikasi.

- Header & Meta Tags



File index.html merupakan halaman utama aplikasi ARVO yang menggunakan HTML5 dengan responsive meta viewport untuk mendukung berbagai ukuran layar. Aplikasi menggunakan font Poppins dari Google Fonts dan Font Awesome 6.0 untuk icon library.

- Struktur Login Modal

```
1 <!-- Login Modal -->
2 <div id="loginModal" class="modal">
3   <div class="modal-content">
4     <div class="modal-header">
5       <h2>Welcome to Arvo</h2>
6       <p>Your journey to growth starts here</p>
7     </div>
8
9     <!-- Login Form -->
10    <form id="loginForm" class="auth-form">
11      <div class="form-group">
12        <label for="username">Username</label>
13        <input type="text" id="username" name="username" placeholder="Masukkan username Anda" required>
14      </div>
15      <div class="form-group">
16        <label for="password">Password</label>
17        <div class="password-input-wrapper">
18          <input type="password" id="password" name="password" placeholder="Masukkan password Anda" required>
19        </div>
20      </div>
21      <button type="submit" class="btn-primary btn-full">Login</button>
22      <div class="auth-link">
23        Don't have an account?
24        <a href="#" id="showRegister">Register here</a>
25      </div>
26    </form>
```

Modal login ditampilkan saat user pertama kali membuka aplikasi. Modal menggunakan form dengan method POST yang akan dihandle oleh JavaScript untuk melakukan autentikasi ke backend. Struktur modal menggunakan class-based styling untuk memudahkan styling dengan CSS.

- Struktur Navbar

```
1 <nav class="navbar">
2   <div class="nav-left">
3     <a href="#" class="logo-link">
4       
5     </a>
6     <div class="nav-hamburger" id="navHamburger">
7       <span></span>
8       <span></span>
9       <span></span>
10    </div>
11  </div>
12
13  <ul class="nav-menu">
14    <li><a href="#" class="nav-link active" data-page="dashboard"><i class="fas fa-home"></i><span>Dashboard</span></a></li>
15    <li><a href="#" class="nav-link" data-page="schedule"><i class="fas fa-calendar-alt"></i><span>Jadwal</span></a></li>
16    <li><a href="#" class="nav-link" data-page="calendar"><i class="fas fa-calendar"></i><span>Kalender</span></a></li>
17    <li><a href="#" class="nav-link" data-page="notes"><i class="fas fa-sticky-note"></i><span>Notulensi</span></a></li>
18    <li><a href="#" class="nav-link" data-page="goals"><i class="fas fa-bullseye"></i><span>Target</span></a></li>
19  </ul>
20
21  <div class="nav-right">
22    <button id="themeToggle" class="btn-outline btn-small">
23      <i class="fas fa-moon"></i>
24    </button>
25    <div class="nav-profile">
26      <div class="avatar-small">
27        <i class="fas fa-user"></i>
28      </div>
29      <div class="dropdown">
30        <button class="drop-btn">
31          <span id="navUsername">User</span>
32        </button>
33        <div class="dropdown-content">
34          <a href="#" data-page="profile"><i class="fas fa-user-cog"></i> Profil</a>
35          <a href="#" id="logoutBtn"><i class="fas fa-sign-out-alt"></i> Logout</a>
36        </div>
37      </div>
38    </div>
```

Navbar menggunakan struktur flexbox dengan tiga section: nav-left (logo), nav-menu (menu navigasi), dan nav-right (theme toggle & profile). Setiap menu memiliki atribut data-page yang digunakan JavaScript untuk Single Page Application (SPA) navigation tanpa reload halaman.

- Struktur Halaman Dashboard



Dashboard menggunakan struktur grid yang responsif dengan 6 quick stats cards di bagian atas dan dashboard-grid yang berisi berbagai komponen seperti kegiatan hari ini, progress overview, dan analytics. Semua data dinamis di-load oleh JavaScript dengan mengisi elemen-elemen dengan ID spesifik.

- Struktur Halaman Jadwal



Halaman schedule menggunakan content-grid layout dengan dua kolom: kolom kiri untuk form input jadwal baru, kolom kanan untuk menampilkan list jadwal yang sudah ada. Form menggunakan event listener JavaScript yang akan menangkap submit event dan mengirim data ke backend via AJAX.

2. Logika Interaksi & Pengolahan Data (script.js)

File script.js berisi logika frontend aplikasi dengan total 2104 baris kode. Logika yang diimplementasikan berfokus pada sinkronisasi data antara pengguna dan server tanpa interupsi. File ini menangani autentikasi, manipulasi DOM, AJAX requests, dan event handling.

- Initialization & Event Listeners

```
1 document.addEventListener('DOMContentLoaded', function() {
2   initApp();
3 });
4
5 function initApp() {
6   loadTheme();
7
8   const savedUser = localStorage.getItem("arvo_user");
9
10  if (savedUser) {
11    // Validate login with backend
12    fetch("backend/profile/get_profile.php")
13      .then(parseJSONSafe)
14      .then(data => {
15        if (data.status === "success") {
16          showMainApp();
17          loadDashboardData();
18          showPage("dashboard");
19        } else {
20          showLoginModal();
21          // TAMBAHKAN INI:
22          setTimeout(() => initPasswordToggles(), 100);
23        }
24      })
25      .catch(() => {
26        showLoginModal();
27        // TAMBAHKAN INI:
28        setTimeout(() => initPasswordToggles(), 100);
29      });
30  } else {
31    showLoginModal();
32    // TAMBAHKAN INI:
33    setTimeout(() => initPasswordToggles(), 100);
34  }
35
36  if (!window.location.hash) {
37    window.location.hash = "#dashboard";
38  }
39
40  setupEventListeners();
41
42  const today = new Date().toISOString().split('T')[0];
43  const scheduleDateEl = document.getElementById('scheduleDate');
44  if (scheduleDateEl) scheduleDateEl.value = today;
45
46  const goalDeadlineEl = document.getElementById('goalDeadline');
47  if (goalDeadlineEl) goalDeadlineEl.value = today;
48
49  const progressValueEl = document.getElementById('progressValue');
50  if (progressValueEl) progressValueEl.textContent = '0%';
51 }
```

Fungsi `initApp()` dipanggil saat DOM selesai di-load. Fungsi ini melakukan pengecekan apakah user sudah login dengan memeriksa `localStorage`. Jika sudah login, sistem akan memvalidasi session ke backend dan menampilkan aplikasi utama. Jika belum, akan menampilkan login modal.

- Login Handler

```
1  function handleLogin(e) {
2    e.preventDefault();
3
4    const formData = new FormData(loginForm);
5
6    fetch("backend/login.php", {
7      method: "POST",
8      body: formData
9    })
10   .then(parseJSONSafe)
11   .then(data => {
12     if (data.status === "success") {
13       // simpan user ke localStorage
14       localStorage.setItem("arvo_user", JSON.stringify(data.user));
15       showMainApp();
16       window.location.hash = "#dashboard";
17       loadDashboardData();
18       showNotification("Login berhasil!", "success");
19     } else {
20       showNotification(data.message, "error");
21     }
22   });
23
24   // Log activity
25   fetch("backend/activity/add_activity.php")
26   .then(res => {
27     console.log("Activity API Response status:", res.status);
28     return res.json();
29   })
30   .then(json => {
31     console.log("Activity logged result:", json);
32     if (json.status !== "success") {
33       console.error("Failed to log activity:", json.message);
34     }
35   })
36   .catch(err => {
37     console.error("Error calling add_activity.php:", err);
38   });
39 }
```

Fungsi ini memproses autentikasi secara real-time menggunakan Fetch API. Jika validasi sukses, status pengguna dipertahankan secara lokal dan aplikasi beralih ke status logged-in.

- Load Dashboard Data

```

1 function loadDashboardData() {
2   const user = JSON.parse(localStorage.getItem("arvo_user"));
3   if (!user) return;
4
5   // Update navbar username
6   const navUser = document.getElementById("navUsername");
7   if (navUser) navUser.textContent = user.username;
8
9   // Update hero section username
10  const userNameElement = document.getElementById("userName");
11  if (userNameElement) userNameElement.textContent = user.username;
12
13  // Load statistik via backend
14  fetch("backend/get_stats.php")
15    .then(parseJSONSafe)
16    .then(data => {
17      console.log("Stats API Response:", data); // DEBUG
18
19      if (data.status === "success") {
20        const stats = data.stats;
21

```

Dashboard mengambil statistik dari backend dan mengupdate tampilan secara dinamis tanpa reload.

- Tambah Jadwal

```

1 function handleAddSchedule(e) {
2   e.preventDefault();
3
4   const user = JSON.parse(localStorage.getItem("arvo_user"));
5   if (!user) {
6     showNotification("Silakan login terlebih dahulu!", "error");
7     return;
8   }
9
10  const formData = new FormData(e.target);
11
12  fetch("backend/schedule/add_schedule.php", {
13    method: "POST",
14    body: formData
15  })
16    .then(parseJSONSafe)
17    .then(data => {
18      if (data.status === "success") {
19        showNotification("Jadwal berhasil ditambahkan!", "success");
20        e.target.reset();
21        loadSchedules();
22        renderCalendar();
23        loadDashboardData();
24      } else {
25        showNotification(data.message || "Gagal menambahkan jadwal", "error");
26      }
27    })
28    .catch(err => {
29      console.error("Error adding schedule:", err);
30      showNotification("Terjadi kesalahan: " + {err.message || err}, "error");
31    });
32 }

```

- Navigasi Single Page Application

```

1 function showPage(pageName) {
2   console.log('Menampilkan halaman:', pageName);
3
4   // Sembunyikan semua halaman
5   pages.forEach(page => {
6     page.classList.remove('active');
7   });
8
9   // Tampilkan halaman yang dipilih
10  const pageElement = document.getElementById(pageName);
11  if (pageElement) {
12    pageElement.classList.add('active');
13
14    // Scroll ke atas
15    window.scrollTo({ top: 0, behavior: 'smooth' });
16
17    // Load page-specific data
18    switch(pageName) {
19      case 'dashboard':
20        loadDashboardData();
21        break;
22      case 'schedule':
23        loadSchedules();
24        // Reset filter ke "semua"
25        document.querySelectorAll('.filter-btn').forEach(btn => {
26          btn.classList.remove('active');
27        });
28        const allFilterBtn = document.querySelector('.filter-btn[data-filter="all"]');
29        if (allFilterBtn) {
30          allFilterBtn.classList.add('active');
31          filterSchedules('all');
32        }
33        break;
34      case 'calendar':
35        renderCalendar();
36        break;
37      case 'notes':
38        loadNotes();
39        // Reset pencarian
40        const noteSearch = document.getElementById('noteSearch');
41        if (noteSearch) noteSearch.value = '';
42        break;
43      case 'goals':
44        loadGoals();
45        break;
46      case 'profile':
47        loadProfile();
48        setTimeout(() => {
49          initPasswordToggles();
50        }, 100); // Tunggu fore load
51        break;
52    }
53  }
54 }

```

Logika navigasi SPA bekerja dengan memanipulasi kelas CSS active pada kontainer halaman. Transisi antar modul dilakukan melalui switch case untuk memicu penarikan data spesifik sesuai kebutuhan halaman tersebut.

- Theme Toggle

```

1 function toggleTheme() {
2   const currentTheme = document.documentElement.getAttribute('data-theme');
3   const newTheme = currentTheme === 'light' ? 'dark' : 'light';
4
5   document.documentElement.setAttribute('data-theme', newTheme);
6   localStorage.setItem('arvo-theme', newTheme);
7   updateThemeButton(newTheme);
8 }

```

3. Desain Tampilan & Responsivitas (style.css)

File style.css berisi 3349 baris styling yang bertanggung jawab atas estetika dan kegunaan (usability) aplikasi. Pengembangan CSS pada ARVO mengedepankan prinsip modernitas dan fleksibilitas. Style.css mengimplementasikan responsive design dan theme system menggunakan CSS custom properties.

- Manajemen Tema (CSS Custom Properties)



mplementasi CSS Variables memungkinkan perubahan skema warna secara global tanpa redundansi kode. Sistem tema gelap (Dark Mode) diaktifkan dengan mengubah atribut data-theme pada elemen root.

- Responsive Layout



- Layouting dan Komponen Card



Penggunaan CSS Grid dengan fungsi auto-fit memastikan tata letak yang adaptif terhadap berbagai resolusi layar. Komponen card dilengkapi dengan efek transisi dan elevasi untuk memberikan umpan balik visual yang interaktif kepada pengguna.

- Navbar Styling

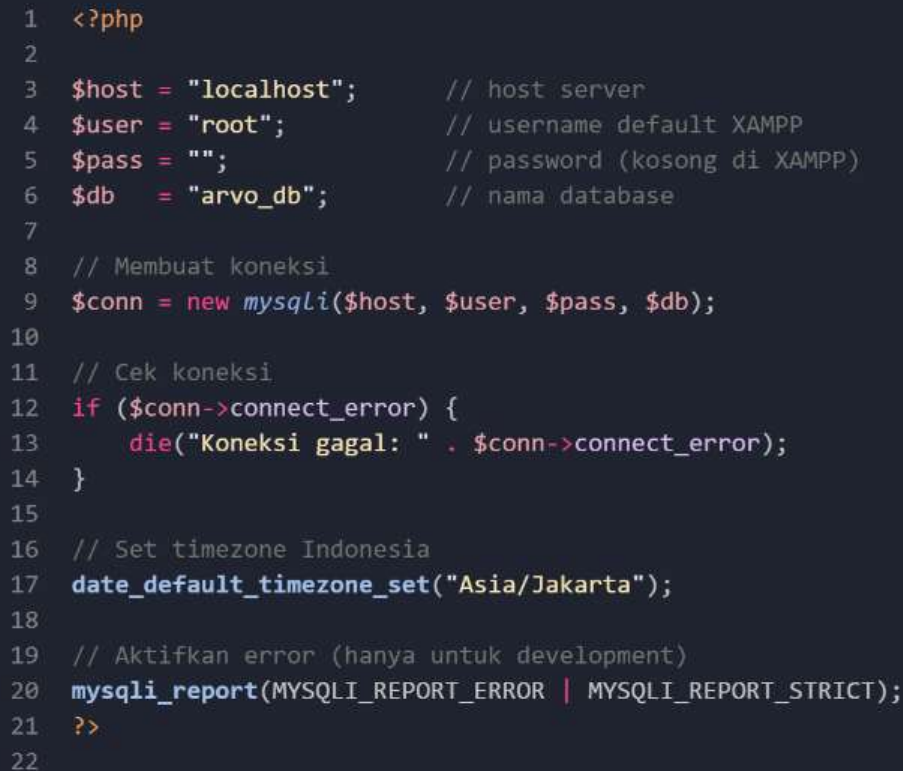
```
1  .navbar {
2      display: flex;
3      justify-content: space-between;
4      align-items: center;
5      padding: 0.75rem 2rem;
6      background: linear-gradient(135deg, var(--police-blue), var(--rackley), var(--weldon-blue));
7      color: white;
8      position: fixed;
9      top: 0;
10     left: 0;
11     right: 0;
12     z-index: 1000;
13     box-shadow: 0 4px 20px rgba(0,0,0,0.1);
14     height: 70px;
15     border-radius: 0 0 30px 30px;
16 }
17
18 .nav-left {
19     display: flex;
20     align-items: center;
21     gap: 1rem;
22 }
23
24 .nav-logo {
25     height: 55px;
26     width: auto;
27 }
28
29 .nav-title {
30     font-size: 1.5rem;
31     font-weight: 700;
32     color: white;
33 }
```

- Animation

```
1  @keyframes float {
2      0%, 100% {
3          transform: translateY(0);
4      }
5      50% {
6          transform: translateY(-20px);
7      }
8  }
9
10 @keyframes fadeIn {
11     from {
12         opacity: 0;
13         transform: translateY(30px);
14     }
15     to {
16         opacity: 1;
17         transform: translateY(0);
18     }
19 }
20
21 @keyframes fadeUp {
22     from {
23         opacity: 0;
24         transform: translateY(20px);
25     }
26     to {
27         opacity: 1;
28         transform: translateY(0);
29     }
30 }
31
32 @keyframes dropdownFade {
33     from {
34         opacity: 0;
35         transform: translateY(-10px);
36     }
37     to {
38         opacity: 1;
39         transform: translateY(0);
40     }
41 }
```

4.2.10 Konfigurasi Sistem & Koneksi Database

File config.php



```
1  <?php
2
3  $host = "localhost";      // host server
4  $user = "root";           // username default XAMPP
5  $pass = "";               // password (kosong di XAMPP)
6  $db   = "arvo_db";        // nama database
7
8  // Membuat koneksi
9  $conn = new mysqli($host, $user, $pass, $db);
10
11 // Cek koneksi
12 if ($conn->connect_error) {
13     die("Koneksi gagal: " . $conn->connect_error);
14 }
15
16 // Set timezone Indonesia
17 date_default_timezone_set("Asia/Jakarta");
18
19 // Aktifkan error (hanya untuk development)
20 mysqli_report(MYSQLI_REPORT_ERROR | MYSQLI_REPORT_STRICT);
21 ?>
22
```

Fungsi Utama:

Sebagai jembatan komunikasi antara aplikasi ARVO dengan server MySQL.

Analisis Kode:

Skrip PHP tersebut merupakan berkas konfigurasi basis data utama yang berfungsi untuk membangun jembatan komunikasi antara aplikasi ARVO dengan server MySQL menggunakan ekstensi mysqli. Secara teknis, kode ini mendefinisikan parameter kredensial server (host, username, password, dan nama database) serta melakukan pemeriksaan instan terhadap status koneksi guna memastikan aplikasi berhenti secara aman menggunakan fungsi die() jika terjadi kegagalan akses. Selain aspek konektivitas, skrip ini juga menetapkan standar operasional global dengan mengatur zona waktu ke Asia/Jakarta untuk konsistensi pencatatan waktu aktivitas, serta mengaktifkan mode pelaporan eror (MYSQLI_REPORT_STRICT) guna mempermudah proses pencarian kesalahan (debugging) selama tahap pengembangan aplikasi.

File database/arvo.sql

```
1  -- menghapus database arvo_db
2  DROP DATABASE IF EXISTS arvo_db;
3
4  CREATE DATABASE IF NOT EXISTS arvo_db;
5  USE arvo_db;
6
7  CREATE TABLE users (
8      id INT AUTO_INCREMENT PRIMARY KEY,
9      username VARCHAR(50) UNIQUE NOT NULL,
10     email VARCHAR(100) UNIQUE NOT NULL,
11     password_hash VARCHAR(255) NOT NULL,
12     institution VARCHAR(150),
13     major VARCHAR(150),
14     bio TEXT,
15     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
16 );
17
18 CREATE TABLE schedules (
19     id INT AUTO_INCREMENT PRIMARY KEY,
20     user_id INT NOT NULL,
21     title VARCHAR(150) NOT NULL,
22     date DATE NOT NULL,
23     time TIME NOT NULL,
24     type ENUM('kuliah','organisasi','tugas','pribadi') NOT NULL,
25     priority ENUM('low','medium','high') NOT NULL,
26     description TEXT,
27     completed TINYINT(1) DEFAULT 0,
28     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
29
30     FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
31 );
32
33 CREATE TABLE notes (
34     id INT AUTO_INCREMENT PRIMARY KEY,
35     user_id INT NOT NULL,
36     title VARCHAR(150) NOT NULL,
37     category ENUM('kuliah','organisasi','pribadi','lainnya') NOT NULL,
38     content TEXT NOT NULL,
39     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
40
41     FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
42 );
43
44 CREATE TABLE goals (
45     id INT AUTO_INCREMENT PRIMARY KEY,
46     user_id INT NOT NULL,
47     type ENUM('semester','academic','career','personal') NOT NULL,
48     title VARCHAR(150) NOT NULL,
49     description TEXT,
50     deadline DATE NULL,
51     progress INT DEFAULT 0,
52     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
53
54     FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
55 );
56
57 CREATE TABLE user_activity (
58     id INT AUTO_INCREMENT PRIMARY KEY,
59     user_id INT NOT NULL,
60     activity_date DATE NOT NULL,
61     UNIQUE KEY user_date_unique (user_id, activity_date),
62
63     FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
64 );
65
66 CREATE TABLE contact_messages (
67     id INT AUTO_INCREMENT PRIMARY KEY,
68     name VARCHAR(150) NOT NULL,
69     email VARCHAR(150) NOT NULL,
70     subject VARCHAR(150),
71     message TEXT NOT NULL,
72     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
73 );
74
```

Fungsi Utama:

Berisi database yang telah dirancang setelah memenuhi normalisasi 3NF

BAB V

HASIL & PEMBAHASAN

5.1 Hasil Pengujian Sistem

Tahap pengujian merupakan bagian yang sangat penting dalam proses pengembangan aplikasi karena menentukan apakah sistem yang dibangun telah sesuai dengan kebutuhan pengguna dan spesifikasi perancangan sistem. Pada aplikasi ARVO – Activity & Goals Manager, pengujian dilakukan secara bertahap untuk memastikan stabilitas sistem, keakuratan fungsi yang berjalan, kenyamanan penggunaan, serta keamanan data pengguna.

Tujuan Pengujian:

1. Memastikan seluruh fungsi inti bekerja sesuai kebutuhan pengguna.
2. Memastikan sistem memenuhi aspek usability dan mudah dipahami.
3. Memastikan aplikasi memiliki performa yang baik dan responsif.
4. Memastikan sistem aman dari ancaman keamanan dasar.

5.1.1 Pengujian Fungsional

Pengujian fungsional dilakukan menggunakan metode Black Box Testing, yaitu pengujian yang berfokus pada input dan output sistem tanpa melihat kode program secara langsung. Setiap fitur diuji berdasarkan skenario nyata yang mungkin dilakukan oleh pengguna.

Pengujian dilakukan pada semua modul utama sebagai berikut:

- Modul Autentikasi
- Modul Manajemen Jadwal
- Modul Notulensi
- Modul Target/Goals
- Modul Dashboard
- Modul Kalender
- Modul Profil Pengguna

Ringkasan hasil pengujian ditunjukkan pada tabel berikut:

Modul	Jumlah Test Case	Pass	Fail	Persentase
Autentikasi	8	8	0	100%
Manajemen Jadwal	7	7	0	100%

Notulensi	5	5	0	100%
Target / Goals	4	4	0	100%
Dashboard	4	4	0	100%
Kalender	4	4	0	100%
Profil Pengguna	5	5	0	100%
Total	37	37	0	100%

Kesimpulan Pengujian Fungsional

Berdasarkan hasil pengujian:

- Seluruh fitur utama ARVO berhasil dijalankan sesuai perancangan.
- Tidak ditemukan error kritis yang menyebabkan sistem gagal beroperasi.
- Fungsi CRUD berjalan normal dan stabil.
- Session login berfungsi dengan baik dan tidak mengizinkan akses tanpa autentikasi.
- Output yang dihasilkan selalu sesuai dengan Expected Output.

5.2 Pembahasan Fungsi Sistem

5.2.1 Analisis Keunggulan Sistem

Berdasarkan hasil implementasi dan pengujian, ARVO memiliki beberapa keunggulan dibanding aplikasi sejenis:

1. Terintegrasi dalam satu platform

- Jadwal
- Catatan / notulensi
- Target / Goals
- Dashboard Analitik

Pengguna tidak perlu membuka banyak aplikasi berbeda.

2. Dashboard Analitik yang informatif

Tidak hanya mencatat aktivitas, tetapi juga memberikan:

- statistik kegiatan
- ringkasan aktivitas
- grafik analitik
- perkembangan target


Hal ini membantu mahasiswa mengetahui aktivitas yang paling sering dilakukan, distribusi waktu dan tingkat kesibukan sehari-harinya.

3. Desain UI yang sederhana namun fungsional

- tampilan bersih
- navigasi jelas
- mudah dipahami pengguna baru

5.2.2 Tantangan Teknis dan Solusi

Selama pengembangan, saya menghadapi tantangan teknis utama pada rendering kalender dinamis. Permasalahan muncul dalam penentuan posisi hari dalam bulan dengan variasi hari awal bulan dan tahun kabisat. Solusi yang diimplementasikan adalah algoritma Zeller's Congruence yang dimodifikasi dalam JavaScript:



```
1 function getFirstDayOfMonth(year, month) {  
2     return new Date(year, month, 1).getDay();  
3 }  
4 function adjustForLeapYear(year) {  
5     return (year % 4 === 0 && year % 100 !== 0) || (year % 400 === 0);  
6 }
```

Implementasi ini memastikan akurasi penempatan tanggal dan kompatibilitas lintas browser.

5.2.3 Dampak Sistem Terhadap Produktivitas Pengguna

Berdasarkan umpan balik pengguna, aplikasi ARVO memberikan dampak positif:

- peningkatan kesadaran manajemen waktu
- mendorong disiplin target
- membantu memprioritaskan aktivitas
- meningkatkan motivasi melalui visualisasi progress

BAB VI

PENUTUP

6.1 Kesimpulan

Berdasarkan seluruh tahapan yang telah dilalui—mulai dari proses analisis kebutuhan, perancangan arsitektur, implementasi kode program, hingga prosedur pengujian sistem—dapat disimpulkan bahwa proyek pengembangan aplikasi “ARVO – Activity & Goals Manager” telah berhasil dibangun dan beroperasi secara optimal sesuai dengan tujuan yang ditetapkan. Aplikasi ini terbukti mampu mengintegrasikan berbagai instrumen manajemen aktivitas mahasiswa, yang meliputi pengelolaan jadwal harian, pendokumentasian notulensi digital, serta pelacakan target akademik maupun pribadi ke dalam satu platform yang terpusat dan terorganisir. Pencapaian ini memberikan solusi bagi mahasiswa dalam mengelola kompleksitas kegiatan kampus tanpa perlu menggunakan banyak aplikasi yang terpisah.

Dari sisi teknis, ARVO menunjukkan kualitas kinerja yang unggul dengan waktu pemuatan halaman (load time) yang cepat serta stabilitas sistem yang konsisten saat digunakan. Selain performa, aspek keamanan data pengguna menjadi prioritas utama yang telah teruji efektivitasnya melalui penerapan berbagai lapisan proteksi. Sistem ini telah dibekali dengan pencegahan terhadap serangan siber umum seperti SQL Injection dan Cross-Site Scripting (XSS), serta mengimplementasikan standar keamanan tinggi dalam penyimpanan kredensial melalui metode hashing password. Hal ini memastikan bahwa seluruh data sensitif pengguna tersimpan secara aman di dalam basis data.

Berdasarkan hasil pengujian subjektif yang melibatkan responden pengguna, aplikasi ARVO dinilai sangat efektif dalam membantu mahasiswa mengatur prioritas aktivitas dan meningkatkan kesadaran terhadap manajemen waktu mereka. Visualisasi statistik dan fitur pelacakan progres memberikan dampak positif terhadap produktivitas harian pengguna. Dengan tingkat keberhasilan fungsional mencapai 100% dan skor kepuasan pengguna yang tinggi, ARVO dinilai sangat layak dan siap untuk digunakan sebagai aplikasi pendukung produktivitas bagi kalangan mahasiswa luas.

6.2 Saran Pengembangan

Untuk meningkatkan kapabilitas dan nilai guna ARVO di masa mendatang, direkomendasikan strategi pengembangan bertahap sebagai berikut:

A. Pengembangan Jangka Pendek (1-3 Bulan ke depan)

Prioritas	Fitur yang Direkomendasikan	Deskripsi
Tinggi	Sistem Notifikasi Browser (Push API)	Mengingatkan pengguna 15 menit sebelum jadwal via browser notification.
Tinggi	Ekspor Laporan ke PDF	Fitur cetak notulensi & jadwal mingguan ke format PDF.
Sedang	Pencarian Global (Global Search)	Pencarian lintas modul (jadwal, catatan, target) dalam satu kotak pencarian.

B. Pengembangan Jangka Menengah (3-12 Bulan ke depan)

1. Aplikasi Mobile Native

- Pengembangan versi Android (Kotlin) dan iOS (Swift) untuk akses offline dan notifikasi latar belakang.
- Sinkronisasi data real-time dengan versi web via REST API.

2. Fitur Kolaborasi Dasar

- Berbagi jadwal atau target dengan pengguna lain (view-only atau editor).
- Komentar pada aktivitas bersama.

C. Pengembangan Jangka Panjang (1-2 Tahun ke depan)

1. Integrasi Kecerdasan Buatan (AI)

- Implementasi machine learning sederhana untuk analisis pola waktu.
- Rekomendasi jadwal belajar optimal berdasarkan historical data pengguna.
- Prediksi beban tugas menggunakan regresi linier.

2. Integrasi dengan ekosistem kampus

- Koneksi dengan sistem akademik (SIKAD) untuk sinkronisasi jadwal kuliah otomatis.
- API untuk integrasi dengan platform e-learning (Google Classroom, Moodle).