

Contents

1 SYSTEM ARCHITECTURE DOCUMENTATION	1
1.1 Application Overview	1
1.2 HIGH-LEVEL ARCHITECTURE	1
1.3 DETAILED ARCHITECTURE COMPONENTS	3
1.3.1 1. CLIENT LAYER	3
1.3.2 2. WEB SERVER LAYER	4
1.3.3 3. APPLICATION LAYER (Laravel MVC)	4
1.3.4 4. DATA LAYER	8
1.4 EXTERNAL INTEGRATIONS	9
1.4.1 Integration Details:	9
1.5 SECURITY ARCHITECTURE	9
1.6 PERFORMANCE OPTIMIZATION	10
1.6.1 1. Frontend Optimization	10
1.6.2 2. Backend Optimization	10
1.6.3 3. Caching Strategy	11
1.7 SCALABILITY ARCHITECTURE	11
1.7.1 Current Setup (Single Server)	11
1.7.2 Scalability Roadmap (Future)	11
1.8 DEPLOYMENT ARCHITECTURE	12
1.8.1 Development Environment	12
1.8.2 Production Environment	12
1.8.3 CI/CD Pipeline (Git-based Deploy)	13
1.9 MONITORING & LOGGING	13
1.9.1 Application Monitoring	13
1.10 TECHNOLOGY STACK SUMMARY	14
1.11 SYSTEM REQUIREMENTS	14
1.11.1 Server Minimum Requirements:	14
1.11.2 PHP Extensions Required:	14
1.12 BACKUP & DISASTER RECOVERY	15
1.13 DOCUMENTATION & SUPPORT	15

1 SYSTEM ARCHITECTURE DOCUMENTATION

1.1 Application Overview

Application Name: Koperasi Karyawan Digital Platform

Architecture Pattern: MVC (Model-View-Controller)

Framework: Laravel 11.x

Deployment Model: Monolithic Web Application

Platform: Web-based (Responsive PWA-ready)

1.2 HIGH-LEVEL ARCHITECTURE

CLIENT LAYER

Daftar Isi

1 ffl DOKUMENTASI ARSITEKTUR SISTEM 1	
1.1 ffl Gambaran Umum Aplikasi	1
1.2 ffl ARSITEKTUR TINGKAT TINGGI	1
1.3 ffl KOMPONEN ARSITEKTUR DETAIL.....	3
1.3.1 1. LAPISAN KLIEN	3
1.3.2 2. LAPISAN SERVER WEB	4
1.3.3 3. LAPISAN APLIKASI (Laravel MVC)	4
1.3.4 4. LAPISAN DATA	8
1.4 ffl INTEGRASI EKSTERNAL	9
1.4.1 Rincian Integrasi:	9
1.5 ffl ARSITEKTUR KEAMANAN	9
1.6 ffl OPTIMISASI KINERJA	10
1.6.1 1. Optimasi Frontend	10
1.6.2 2. Optimasi Backend	10
1.6.3 3. Strategi Caching	11
1.7 ffl ARSITEKTUR SKALABILITAS	11
1.7.1 Pengaturan Saat Ini (Server Tunggal)	11
1.7.2 Peta Jalan Skalabilitas (Masa Depan)	11
1.8 ffl ARSITEKTUR DEPLOYMENT	12
1.8.1 Lingkungan Pengembangan	12
1.8.2 Lingkungan Produksi	12
1.8.3 Pipeline CI/CD (Deploy Berbasis Git)	13
1.9 ffl MONITORING & LOGGING	13
1.9.1 Pemantauan Aplikasi	13
1.10 ffl RINGKASAN TUMPUKAN TEKNOLOGI	14
1.11 ffl PERSYARATAN SISTEM	14
1.11.1 Persyaratan Minimum Server:	14
1.11.2 Ekstensi PHP yang Diperlukan:	14
1.12 ffl CADANGAN & PEMULIHAN BENCANA	15
1.13 DOKUMENTASI & DUKUNGAN	15

1 DOKUMENTASI ARSITEKTUR SISTEM

1.1 Gambaran Umum Aplikasi

Nama Aplikasi: Koperasi Karyawan Platform DigitalPola

Arsitektur: MVC (Model-View-Controller)Kerangka:

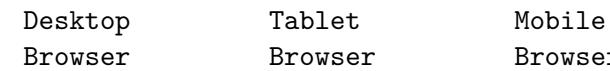
Laravel 11.x

Model Penempatan: Aplikasi Web MonolitikPlatform:

Berbasis Web (Siap PWA Responsif)

1.2 ARSITEKTUR TINGKAT TINGGI

LAPISAN KLIEN



HTTPS (SSL/TLS)

WEB SERVER LAYER

Nginx / Apache HTTP Server

- SSL Termination
 - Static Asset Serving
 - Reverse Proxy to PHP-FPM
 - Gzip Compression

APPLICATION LAYER (Laravel)

ROUTING LAYER

MIDDLEWARE LAYER

- Authentication (Sanctum)
 - Authorization (Policy, Gates)
 - CSRF Protection
 - Rate Limiting
 - Session Management

CONTROLLER LAYER (MVC)

Member Finance Commerce
Controller Controller Controller

Desktop
Browser

TabletB
rowser

Desktop
Browser
Browser
Browser

HTTPS (SSL/TLS)

LAPIS SERVER WEB

Nginx / Apache HTTP Server

- Terminasi SSL
- Penyajian Aset Statis
- Proxy Balik ke PHP-FPM
- Kompresi Gzip

LAPIS APLIKASI (Laravel)

LAPIS RUTE

web.php
(Rute Web)

api.php
(Rute API)

LAPIS MIDDLEWARE

- Autentikasi (Sanctum)
- Otorisasi (Kebijakan, Gerbang)
- Perlindungan CSRF
- Pembatasan Laju
- Manajemen Sesi

LAPISAN KONTROLER (MVC)

Anggota
Pengontrol

Keuangan
Pengontrol

Perdagangan
Pengontrol

BUSINESS LOGIC LAYER

- Validation
- Data Processing
- Business Rules Enforcement

MODEL LAYER (Eloquent ORM)

Member Loan Product

Model Model Model

...

- Relationships
- Accessors & Mutators
- Eloquent Scopes

VIEW LAYER (Blade)

- Template Rendering
- Alpine.js (Frontend Reactivity)
- Tailwind CSS (Styling)

DATABASE STORAGE CACHE
LAYER LAYER LAYER
(MySQL 8.0) (Files) (File/Redis)

1.3 DETAILED ARCHITECTURE COMPONENTS

1.3.1 1. CLIENT LAYER

Supported Browsers: - Chrome 90+ - Firefox 88+ - Safari 14+ - Edge 90+

Technologies: - **HTML5** - Semantic markup - **CSS3** - Tailwind CSS framework - **JavaScript (ES6+)** - Alpine.js for reactivity - **Responsive Design** - Mobile-first approach

Features: - Progressive Web App (PWA) ready - Offline capability (service workers) - Push notifications support

LAPIS LOGIKA BISNIS

- Validasi
- Pemrosesan Data
- Penegakan Aturan Bisnis

LAPIS MODEL (Eloquent ORM)



- Hubungan
- Akses dan Mutator
- Lingkup Eloquent

LAPISAN TAMPILAN (Blade)

- Pembuatan Template
- Alpine.js (Reaktivitas Frontend) -
- Tailwind CSS (Styling)



1.3 KOMPONEN ARSITEKTUR DETAIL

1.3.1 1. LAPISAN KLIEN

Browser yang Didukung: - Chrome 90+ - Firefox 88+ - Safari 14+ - Edge 90+

Teknologi: - HTML5 - Penandaan semantik - CSS3 - Kerangka kerja Tailwind CSS - JavaScript (ES6+) - Alpine.js untuk reaktivitas - Desain Responsif - Pendekatan mobile-first

Fitur: - Aplikasi Web Progresif (PWA) siap - Kemampuan offline (pekerja layanan) - Dukungan notifikasi push

1.3.2 2. WEB SERVER LAYER

Nginx Configuration (Production):

```
server {
    listen 80;
    server_name kopkarskf.com;
    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl http2;
    server_name kopkarskf.com;
    root /var/www/koperasi/public;

    ssl_certificate /etc/letsencrypt/live/kopkarskf.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/kopkarskf.com/privkey.pem;

    index index.php index.html;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location ~ \.php$ {
        fastcgi_pass unix:/var/run/php/php8.2-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }

    # Static assets caching
    location ~* \.(jpg|jpeg|png|gif|ico|css|js|svg|woff|woff2)$ {
        expires 365d;
        add_header Cache-Control "public, immutable";
    }
}
```

1.3.3 3. APPLICATION LAYER (Laravel MVC)

1.3.3.1 A. Routing Layer

```
// routes/web.php
Route::middleware(['auth', 'verified'])->group(function () {
    Route::get('/dashboard', [DashboardController::class, 'index'])
        ->name('dashboard');
```

1.3.2 2. LAPISAN SERVER WEB

Konfigurasi Nginx (Produksi):

```
server {  
    listen 80;  
    server_name kopkarskf.com;  
    return 301 https://$server_name$request_uri;  
}  
  
server {  
    listen 443 ssl http2;  
    server_name kopkarskf.com;  
    root /var/www/koperasi/public;  
  
    ssl_certificate /etc/letsencrypt/live/kopkarskf.com/fullchain.pem;  
    ssl_certificate_key /etc/letsencrypt/live/kopkarskf.com/privkey.pem;  
  
    index index.php index.html;  
  
    location / {  
        try_files $uri $uri/ /index.php?$query_string;  
    }  
  
    location ~ \.php$ {  
        fastcgi_pass unix:/var/run/php/php8.2-fpm.sock;  
        fastcgi_index index.php;  
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;  
        include fastcgi_params;  
    }  
  
    # Cache asset statis  
    location ~* \.(jpg|jpeg|png|gif|ico|css|js|svg|woff|woff2)$ {  
        expires 365d;  
        add_header Cache-Control "public, immutable";  
    }  
}
```

1.3.3 3. LAPIS APLIKASI (Laravel MVC)

1.3.3.1 A. Lapisan Routing

```
// routes/web.php  
Route::middleware(['auth', 'verified'])->group(function () {  
    Route::get('/dashboard', [DashboardController::class, 'index'])  
        ->name('dashboard');
```

```

Route::resource('members', MemberController::class);
Route::resource('loans', LoanController::class);
Route::resource('products', ProductController::class);
// ... more routes
});

```

Route Groups: - auth - Authenticated users only - verified - Email verified users - role:admin
- Admin-only access - throttle - Rate limiting

1.3.3.2 B. Middleware Stack

```

// app/Http/Kernel.php
protected $middlewareGroups = [
    'web' => [
        \App\Http\Middleware\EncryptCookies::class,
        \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
        \Illuminate\Session\Middleware\StartSession::class,
        \Illuminate\View\Middleware\ShareErrorsFromSession::class,
        \App\Http\Middleware\VerifyCsrfToken::class,
        \Illuminate\Routing\Middleware\SubstituteBindings::class,
        \App\Http\Middleware\GlobalSettings::class, // Custom
    ],
];

```

Custom Middleware: - GlobalSettings - Load koperasi settings for all requests - RoleMiddleware - Role-based access control - AuditMiddleware - Log user activities

1.3.3.3 C. Controller Layer Controller Structure:

```

app/Http/Controllers/
Auth/
    LoginController.php
    RegisterController.php
    MemberController.php
    SavingController.php
    LoanController.php
    TransactionController.php
    ProductController.php
    JournalController.php
    DocumentController.php
    ... (50+ controllers)

```

Example Controller Pattern:

```

namespace App\Http\Controllers;

class LoanController extends Controller
{
    public function index()

```

```

Route::resource('members', MemberController::class);
Route::resource('loans', LoanController::class);
Route::resource('products', ProductController::class);
// ... lebih banyak rute
});

```

Route Groups: - auth - Hanya pengguna yang terautentikasi - verified - Pengguna yang telah memverifikasi email - role:admin- Akses hanya untuk admin - throttle - Pembatasan laju

1.3.3.2 B. Tumpukan Middleware

```

// app/Http/Kernel.php
protected $middlewareGroups = [
    'web' => [
        \App\Http\Middleware\EncryptCookies::class,
        \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
        \Illuminate\Session\Middleware\StartSession::class, \Illuminate\View\Middleware\ShareErrorsFromSession::class, \App\Http\Middleware\VerifyCsrfToken::class, \Illuminate\Routing\Middleware\SubstituteBindings::class, \App\Http\Middleware\GlobalSettings::class, // Kustom
    ]
];

```

Middleware Kustom: - GlobalSettings - Memuat pengaturan koperasi untuk semua permintaan - RoleMiddleware - kontrol akses berbasis peran - AuditMiddleware - Mencatat aktivitas pengguna

1.3.3.3 C. Lapisan Kontrol Struktur Kontroler:

```

app/Http/Controllers/
Auth/
    LoginController.php
    RegisterController.php
    MemberController.php
    SavingController.php
    LoanController.php
    TransactionController.php
    ProductController.php
    JournalController.php
    DocumentController.php
    ... (50+ pengontrol)

```

Contoh Pola Pengontrol: namespace

```
App\Http\Controllers;
```

```
class LoanController extends Controller{
```

```
    public function index()
```

```

{
    // Authorization check
    $this->authorize('viewAny', Loan::class);

    // Business logic
    $loans = Loan::with(['member', 'payments'])
        ->when(request('status'), function($q, $status) {
            $q->where('status', $status);
        })
        ->paginate(20);

    // Return view
    return view('loans.index', compact('loans'));
}

public function store(Request $request)
{
    // Validation
    $validated = $request->validate(...);

    // Create loan with transaction
    DB::transaction(function() use ($validated) {
        $loan = Loan::create($validated);
        $loan->generatePayments();
        $loan->createJournalEntry();
    });

    // Notification
    event(new LoanCreated($loan));

    return redirect()->route('loans.index');
}
}

```

1.3.3.4 D. Model Layer (Eloquent ORM) Model Structure:

```

app/Models/
User.php
Member.php
Saving.php
Loan.php
LoanPayment.php
Product.php
Transaction.php
JournalEntry.php
... (60+ models)

```

Example Model:

```

{
    // Pemeriksaan otorisasi
    $this->authorize('viewAny', Loan::class);

    // Logika bisnis
    $loans = Loan::with(['member', 'payments'])
        ->when(request('status'), function($q, $status) {
            $q->where('status', $status);
        })
        ->paginate(20);

    // Kembalikan tampilan
    return view('loans.index', compact('loans'));
}

public function store(Request $request){

    // Validasi
    $validated = $request->validate([...]);

    // Buat pinjaman dengan transaksi
    DB::transaction(function() use ($validated) {
        $loan = Loan::create($validated);
        $loan->generatePayments();
        $loan->createJournalEntry();
    });

    // Pemberitahuan
    event(new LoanCreated($loan));

    return redirect()->route('loans.index');
}
}

```

1.3.3.4 D. Lapisan Model (Eloquent ORM) Struktur Model:

```

app/Models/
User.php
Member.php
Saving.php
Loan.php
    LoanPayment.php
Product.php
    Transaction.php
JournalEntry.php
...
... (60+ model)

```

Contoh Model:

```

namespace App\Models;

class Loan extends Model
{
    protected $fillable = [...];
    protected $casts = [
        'amount' => 'decimal:2',
        'approved_at' => 'datetime',
    ];

    // Relationships
    public function member() {
        return $this->belongsTo(Member::class);
    }

    public function payments() {
        return $this->hasMany(LoanPayment::class);
    }

    // Accessors
    public function getRemainingBalanceAttribute() {
        return $this->amount - $this->payments->sum('amount');
    }

    // Business Logic
    public function generatePayments() {
        // Complex calculation...
    }
}

```

1.3.3.5 E. View Layer (Blade Templates) View Structure:

```

resources/views/
    layouts/
        app.blade.php (Master template)
        sidebar.blade.php
        navbar.blade.php
        dashboard.blade.php
    members/
        index.blade.php
        create.blade.php
        show.blade.php
    loans/
    products/
    ... (100+ blade files)

```

Technology Stack: - **Blade** - Laravel templating engine - **Alpine.js** - Lightweight reactive framework - **Tailwind CSS** - Utility-first CSS - **Chart.js** - Data visualization

```

namespace App\Models;

class Loan extends Model
{
    protected $fillable = [...] ;
    protected $casts = [
        'amount' => 'decimal:2',
        'approved_at' => 'datetime',
    ] ;

    // Hubungan
    public function member() {
        return $this->belongsTo(Member::class);
    }

    public function payments() {
        return $this->hasMany(LoanPayment::class);
    }

    // Accessors
    public function getRemainingBalanceAttribute() {
        return $this->amount - $this->payments->sum('amount');
    }

    // Business Logic
    public function generatePayments() {
        // Perhitungan kompleks...
    }
}

```

1.3.3.5 E. Lapisan Tampilan (Template Blade) Struktur Tampilan:

```

resources/views/
layouts/
    app.blade.php (Template utama)
    sidebar.blade.php
    navbar.blade.php
    dashboard.blade.php
members/
    index.blade.php
    create.blade.php
    show.blade.php
loans/
    products/
    ... (100+ file blade)

```

Tumpukan Teknologi: - Blade - mesin templating Laravel - Alpine.js - kerangka reaktif ringan - Tailwind CSS - CSS utilitas pertama - Chart.js - visualisasi data

1.3.4 4. DATA LAYER

1.3.4.1 A. Database (MySQL) Connection Pool:

```
// config/database.php
'mysql' => [
    'driver' => 'mysql',
    'host' => env('DB_HOST', '127.0.0.1'),
    'port' => env('DB_PORT', '3306'),
    'database' => env('DB_DATABASE', 'koperasi'),
    'username' => env('DB_USERNAME', 'root'),
    'password' => env('DB_PASSWORD', ''),
    'charset' => 'utf8mb4',
    'collation' => 'utf8mb4_unicode_ci',
    'strict' => true,
    'engine' => 'InnoDB',
],
```

Query Optimization: - Eager loading (N+1 prevention) - Index on foreign keys - Query result caching

1.3.4.2 B. File Storage Storage Structure:

```
storage/app/public/
members/           → Member photos
products/          → Product images
documents/         → Uploaded docs
logos/             → Koperasi logos
announcements/    → Announcement images
backups/           → Database backups
```

Storage Configuration:

```
// config/filesystems.php
'public' => [
    'driver' => 'local',
    'root' => storage_path('app/public'),
    'url' => env('APP_URL').'/storage',
    'visibility' => 'public',
],
```

1.3.4.3 C. Caching Layer Cache Strategy: - **Config Cache:** php artisan config:cache
- **Route Cache:** php artisan route:cache - **View Cache:** php artisan view:cache - **Query Cache:** Result caching for expensive queries

```
// Example cached query
$members = Cache::remember('active_members', 3600, function() {
    return Member::where('status', 'active')->get();
});
```

1.3.4 4. LAPISAN DATA

1.3.4.1 A. Koneksi Pool Database (MySQL):

```
// config/database.php
'mysql' => [
    'driver' => 'mysql',
    'host' => env('DB_HOST', '127.0.0.1'),
    'port' => env('DB_PORT', '3306'),
    'database' => env('DB_DATABASE', 'koperasi'),
    'username' => env('DB_USERNAME', 'root'),
    'password' => env('DB_PASSWORD', ''),
    'charset' => 'utf8mb4',
    'collation' => 'utf8mb4_unicode_ci',
    'strict' => true,
    'engine' => 'InnoDB',
],
```

Optimasi Kuery: - Pemuatan cepat (pencegahan N+1) - Indeks pada kunci asing - Penyimpanan hasil kuery

1.3.4.2 B. Struktur Penyimpanan File:

storage/app/public/	
anggota/	→ Foto anggota→ Gambar
produk/	produk→ Dokumen yang
dokumen/	diunggah→ Logo
logo/	koperasi→ Gambar
pengumuman/	pengumuman→
cadangan/	Cadangan database

Konfigurasi Penyimpanan:

```
// config/filesystems.php
'publik' => [
    'driver' => 'local',
    'root' => storage_path('app/public'),
    'url' => env('APP_URL') . '/storage',
    'visibility' => 'publik',
],
```

1.3.4.3 C. Strategi Cache Layer Cache: - Cache Konfigurasi: php artisan config:cache
- Cache Rute: php artisan route:cache - Cache Tampilan: php artisan view:cache - QueryCache: Cache hasil untuk query yang mahal

```
// Contoh query yang dicache
$members = Cache::remember('active_members', 3600, function() {
    return Member::where('status', 'active')->get();
});
```

1.4 EXTERNAL INTEGRATIONS

LARAVEL APPLICATION

Midtrans Payment Gateway	SMTP Email	WhatsApp API	QR Code Server (Fonnte)	Google Drive Backup
--------------------------	------------	--------------	-------------------------	---------------------

1.4.1 Integration Details:

Service	Purpose	Protocol	Config Location
Midtrans	Payment processing	REST API	.env (MIDTRANS_*)
SMTP (Gmail)	Email notifications	SMTP	.env (MAIL_*)
Fonnte/Twilio	WhatsApp messaging	REST API	.env (WHATSAPP_*)
QR Server	QR code generation	HTTP GET	Hardcoded URL
Google Drive	Cloud backup	OAuth 2.0	config/backup.php

1.5 SECURITY ARCHITECTURE

SECURITY LAYERS

1. TRANSPORT SECURITY

HTTPS/TLS 1.3
SSL Certificate (Let's Encrypt)

2. APPLICATION SECURITY

CSRF Token Protection
XSS Prevention (Blade escaping)
SQL Injection Prevention (Eloquent)
Password Hashing (bcrypt)
Rate Limiting (throttle middleware)

3. AUTHENTICATION & AUTHORIZATION

Laravel Sanctum (Session-based)
Role-Based Access Control (RBAC)

1.4 INTEGRASI EKSTERNAL

APLIKASI LARAVEL

Midtrans	SMTP	WhatsApp	Kode QR	Google
Pembayaran	Email	API	Server	Drive
Gerbang	(Fonnte)			Cadangan

1.4.1 Rincian Integrasi:

Layanan	Tujuan	Protokol	Lokasi Konfigurasi
Midtrans	Pemrosesan pembayaran	REST API	.env (MIDTRANS_*) .env
Notifikasi email	SMTP (Gmail)	Notifikasi email SMTP (Gmail) SMTP	(MAIL_*) .env
Fonnte/Twilio	API REST pesan WhatsApp		(WHATSAPP_*)
Server QR	Generasi kode QR	HTTP GET URL yang dikodekan keras	
Google Drive	Cadangan cloud	OAuth 2.0	config/backup.php

1.5 ARSITEKTUR KEAMANAN

LAPISAN KEAMANAN

1. KEAMANAN TRANSPORTASI

HTTPS/TLS 1.3
Sertifikat SSL (Let's Encrypt)

2. KEAMANAN APLIKASI

Perlindungan Token CSRF
Pencegahan XSS (pelarian Blade)
Pencegahan SQL Injection (Eloquent)
Hashing Kata Sandi (bcrypt)
Pembatasan Laju (middleware throttle)

3. AUTENTIKASI & OTORISASI

Laravel Sanctum (berbasis sesi)
Kontrol Akses Berbasis Peran (RBAC)9

Policies & Gates
Multi-factor Authentication (planned)

4. DATA SECURITY

Encrypted Sensitive Fields
Audit Logging (all CRUD operations)
Daily Encrypted Backups
GDPR Compliance Ready

Security Best Practices: - All forms protected with CSRF token - User input sanitized & validated - Passwords never stored plain-text (bcrypt) - File uploads: type & size validation - API rate limiting: 60 requests/minute - Session timeout: 120 minutes

1.6 PERFORMANCE OPTIMIZATION

1.6.1 1. Frontend Optimization

ASSETS PIPELINE:

Source Files (CSS, JS, Images)

Laravel Mix (Asset compilation)
- Minify CSS
- Minify JS
- Image Opt

public/build/ (Optimized assets)

Techniques: - CSS/JS minification - Image lazy loading - WebP image format - Gzip compression - Browser caching (365 days for static)

1.6.2 2. Backend Optimization

Query Optimization:

```
// Bad: N+1 Problem
$members = Member::all();
foreach($members as $member) {
    echo $member->user->name; // N queries
}
```

Kebijakan & Gerbang
Autentikasi Multi-faktor (direncanakan)

4. KEAMANAN DATA

Bidang Sensitif Terenkripsi
Pencatatan Audit (semua operasi
CRUD) Cadangan Terenkripsi Harian
Siap Mematuhi GDPR

Praktik Terbaik Keamanan: - Semua formulir dilindungi dengan token CSRF - Input pengguna disanitasi & dvalidasi - Kata sandi tidak pernah disimpan dalam teks biasa (bcrypt) - Unggahan file: validasi jenis & ukuran - Pembatasan laju API: 60 permintaan/menit - Waktu habis sesi: 120 menit

1.6 OPTIMISASI KINERJA

1.6.1 Optimasi Frontend

SALURAN ASET:

File Sumber (CSS, JS, Gambar)

Laravel Mix (Kompilasi asset)

- Minify CSS -
Minify JS-
Optimasi Gambar

publik/build/ (aset yang dioptimalkan)

Teknik: - minifikasi CSS/JS - pemuatan gambar malas - format gambar WebP - kompresi Gzip- caching browser (365 hari untuk statis)

1.6.2 Optimasi Backend

Optimasi Query:

```
// Buruk: Masalah N+1
$members = Member::all();
foreach($members as $member) {
    echo $member->user->name; // N queries
}
```

```
// Good: Eager Loading
$members = Member::with('user')->get();
foreach($members as $member) {
    echo $member->user->name; // 2 queries only
}
```

Indexing: - All foreign keys indexed - Composite indexes for common queries - Unique constraints for business keys

1.6.3 3. Caching Strategy

```
// Configuration cache (production only)
php artisan config:cache
php artisan route:cache
php artisan view:cache

// Query result cache
Cache::remember('dashboard_stats', 600, function() {
    return [
        'total_members' => Member::count(),
        'total_savings' => Saving::sum('amount'),
        // ... expensive queries
    ];
});
```

1.7 SCALABILITY ARCHITECTURE

1.7.1 Current Setup (Single Server)

SINGLE VPS SERVER

Nginx + PHP-FPM + MySQL
 RAM: 4GB
 CPU: 2 cores
 Storage: 50GB SSD

Capacity: ~100 concurrent users

1.7.2 Scalability Roadmap (Future)

Stage 1: Vertical Scaling - Upgrade VPS (4→8GB RAM, 2→4 CPU cores) - Capacity: ~500 concurrent users

Stage 2: Horizontal Scaling

```
// Baik: Eager Loading
$members = Member::with('user')->get();
foreach($members as $member) {
echo $member->user->name; // 2 kueri saja}
```

Pengindeksan: - Semua kunci asing diindeks - Indeks komposit untuk kueri umum - Pembatasan unik untuk kunci bisnis

1.6.3 3. Strategi Cache

```
// Cache konfigurasi (hanya produksi)
php artisan config:cachephp
artisan route:cachephp
artisan view:cache
```

// Cache hasil kueri

```
Cache::remember('dashboard_stats', 600, function() {
    return [
        'total_members' => Member::count(),
        'total_savings' => Saving::sum('amount'),
        // ... kueri mahal
    ];
});
```

1.7 ARSITEKTUR SKALABILITAS

1.7.1 Pengaturan Saat Ini (Server Tunggal)

SERVER VPS TUNGGAL

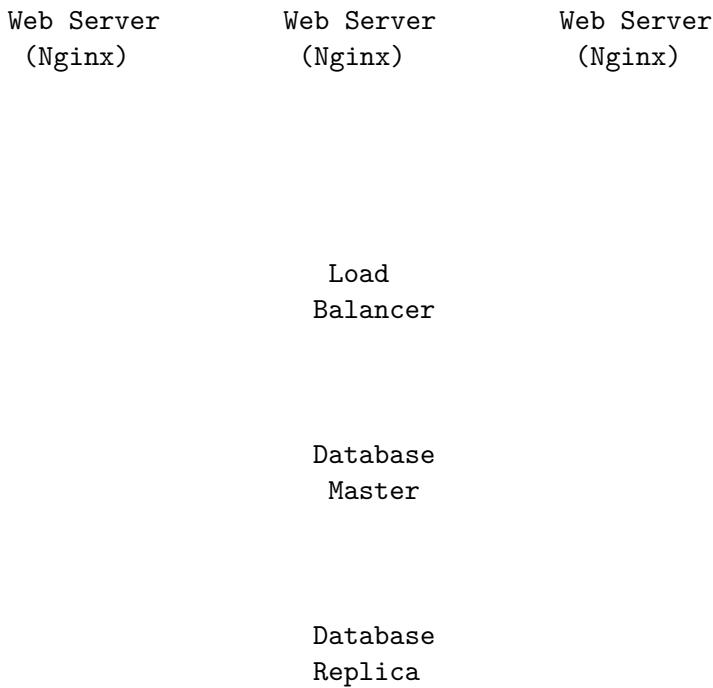
Nginx + PHP-FPM +
MySQLRAM: 4GB
CPU: 2 inti
Penyimpanan: 50GB SSD

Kapasitas: ~100 pengguna bersamaan

1.7.2 Peta Jalan Skalabilitas (Masa Depan)

Tahap 1: Skalabilitas Vertikal - Upgrade VPS (4→8GB RAM, 2→4 inti CPU) - Kapasitas: ~500 pengguna bersamaan

Tahap 2: Skalabilitas Horizontal



Stage 3: Microservices (Long-term) - Separate service for POS transactions - Separate service for reporting - Message queue (Redis/RabbitMQ) - API Gateway

1.8 DEPLOYMENT ARCHITECTURE

1.8.1 Development Environment

- LARAGON (Windows)
 - PHP 8.2
 - MySQL 8.0
 - Apache/Nginx
 - Redis (optional)

1.8.2 Production Environment

VPS (Ubuntu 22.04 LTS)

Application Stack

- Nginx 1.24
- PHP 8.2 (FPM)
- MySQL 8.0
- Redis 7.0 (cache & sessions)
- Supervisor (queue workers)

Server Web
(Nginx) Server Web
(Nginx) Server Web
(Nginx)

Beban
Penyeimbang

Database
Utama

Database
Salinan

Tahap 3: Microservices (Jangka Panjang) - Layanan terpisah untuk transaksi POS - Layanan terpisah untuk pelaporan - Antrian pesan (Redis/RabbitMQ) - API Gateway

1.8 ARSITEKTUR DEPLOYMENT

1.8.1 Lingkungan Pengembangan

LARAGON (Windows)
- PHP 8.2-
MySQL 8.0
- Apache/Nginx
- Redis (opsional)

1.8.2 Lingkungan Produksi

VPS (Ubuntu 22.04 LTS)

Tumpukan Aplikasi
Nginx 1.24
PHP 8.2 (FPM)
MySQL 8.0
Redis 7.0 (cache & sessions)
Supervisor (queue workers)

Security:
UFW Firewall
Fail2ban
SSL via Let's Encrypt

Backup:
Daily cron backup to Google Drive

1.8.3 CI/CD Pipeline (Git-based Deploy)

Local Dev

```
git push
```

GitHub/GitLab

```
git pull (on server)
```

Production Server

```
composer install
php artisan migrate
php artisan config:cache
php artisan route:cache
sudo systemctl reload php8.2-fpm
```

1.9 MONITORING & LOGGING

1.9.1 Application Monitoring

MONITORING STACK

1. Application Logs
`storage/logs/laravel.log`
2. Audit Logs
Database: `audit_logs` table
3. Error Tracking (Recommended)
`Sentry / Bugsnag`
4. Uptime Monitoring

Keamanan:

- UFW Firewall
- Fail2ban
- SSL melalui Let's Encrypt

Cadangan:

- Cadangan cron harian ke Google Drive

1.8.3 CI/CD Pipeline (Deploy berbasis Git)

Pengembangan
Lokal

git push

GitHub/GitLab

git pull (di server)

Server Produksi

```
composer install php
artisan migrate
php artisan config:cache
php artisan route:cache
sudo systemctl reload php8.2-fpm
```

1.9 MONITORING & LOGGING

1.9.1 Pemantauan Aplikasi

STACK PEMANTAUAN

1. Log Aplikasi

storage/logs/laravel.log

2. Log Audit

Database: tabel audit_logs

3. Pelacakan Kesalahan (Disarankan)

Sentry / Bugsnag

4. Pemantauan Waktu Aktif

UptimeRobot / Pingdom

5. Analytics

Google Analytics

Log Rotation:

```
# Laravel daily log rotation
'daily' => [
    'driver' => 'daily',
    'path' => storage_path('logs/laravel.log'),
    'level' => env('LOG_LEVEL', 'debug'),
    'days' => 14,
],
```

1.10 TECHNOLOGY STACK SUMMARY

Layer	Technology	Version
Backend	PHP	8.2+
Framework	Laravel	11.x
Database	MySQL	8.0+
Cache	File/Redis	7.0+
Web Server	Nginx	1.24+
Frontend	Alpine.js	3.x
CSS	Tailwind CSS	3.x
PDF	DomPDF	2.x
Payments	Midtrans SDK	Latest
Version Control	Git	2.x
Deployment	Manual/Forge	-

1.11 SYSTEM REQUIREMENTS

1.11.1 Server Minimum Requirements:

- **OS:** Ubuntu 20.04+ / Debian 11+
- **PHP:** 8.2+
- **MySQL:** 8.0+ / MariaDB 10.6+
- **RAM:** 2GB minimum, 4GB recommended
- **Storage:** 50GB SSD
- **Bandwidth:** 100GB/month

1.11.2 PHP Extensions Required:

- OpenSSL

UptimeRobot / Pingdom

5. Analitik Google Analytics

Rotasi Log:

```
# Rotasi log harian Laravel
'daily' => [
    'driver' => 'daily',
    'path' => storage_path('logs/laravel.log'),
    'level' => env('LOG_LEVEL', 'debug'),
    'days' => 14,
],
```

1.10 RINGKASAN TUMPUK TEKNOLOGI

Lapisan	Teknologi	Versi
Backend	PHP	8.2+
Kerangka kerja	Laravel	11.x
Basis data	MySQLFile/	8.0+
Cache	RedisNginxA	7.0+
Server Web	Ipine.jsTailwi	1.24+
Frontend	nd CSS	3.xTailwind
CSS		CSS3.x
PDF	DomPDF	2.x
Pembayaran	Midtrans SDK Terbaru	
Kontrol Versi Git		2.x
Penerapan	Manual/Forge	-

1.11 SYARAT SISTEM

1.11.1 Syarat Minimum Server:

- OS: Ubuntu 20.04+ / Debian 11+
- PHP: 8.2+
- MySQL: 8.0+ / MariaDB 10.6+
- RAM: 2GB minimum, 4GB recommended
- Penyimpanan: 50GB SSD
- Bandwidth: 100GB/bulan

1.11.2 Ekstensi PHP yang Diperlukan:

- OpenSSL

- PDO (MySQL)
 - Mbstring
 - Tokenizer
 - XML
 - Ctype
 - JSON
 - BCMath
 - GD / Imagick
 - Zip
-

1.12 BACKUP & DISASTER RECOVERY

BACKUP ARCHITECTURE

Daily Backup (Automated - 02:00 AM)

1. Database Dump (SQL)
2. File System (storage/)
3. .env file

Encrypt with GPG

Upload to Google Drive

Retention Policy:

- Daily: 30 days
- Weekly: 3 months
- Monthly: 1 year

Recovery Time Objective (RTO): < 4 hours

Recovery Point Objective (RPO): < 24 hours

1.13 DOCUMENTATION & SUPPORT

Architecture Documentation: - System Architecture: ARCHITECTURE.md (this file) - Database Schema: DATABASE_SCHEMA.md - Features List: FEATURES.md - Deployment Guide: DEPLOYMENT.md

- PDO (MySQL)
 - Mbstring
 - Tokenizer
 - XML
 - Ctype
 - JSON
 - BCMath
 - GD / Imagick
 - Zip
-

1.12 CADANGAN & PEMULIHAN BENCANA

ARSITEKTUR CADANGAN

Cadangan Harian (Otomatis - 02:00 AM)

1. Dump Basis Data (SQL)
2. Sistem Berkas (storage/)
3. Berkas .env

Enkripsi dengan GPG

Unggah ke Google Drive

Kebijakan Retensi:

- Harian: 30 hari
- Mingguan: 3 bulan
- Bulanan: 1 tahun

Tujuan Waktu Pemulihan (RTO): < 4 jam

Titik Pemulihan (RPO): < 24 jam

1.13 DOKUMENTASI & DUKUNGAN

Dokumentasi Arsitektur: - Arsitektur Sistem: ARCHITECTURE.md (file ini) - Skema Database: DATABASE_SCHEMA.md - Daftar Fitur: FEATURES.md - Panduan Penyebaran: DEPLOYMENT.md

- API Reference: `API_DOCUMENTATION.md` (if needed)

Technical Support: - Email: dev@kopkarskf.com - Issue Tracker: GitHub Issues (if open-source)

Document Version: 1.0

Last Updated: 17 January 2026

Maintainer: Architecture Team

Review Cycle: Quarterly

- Referensi API: API_DOCUMENTATION.md (jika diperlukan)

Dukungan Teknis: - Email: dev@kopkarskf.com - Pelacak Masalah: Isu GitHub (jika sumber terbuka)

Versi Dokumen: 1.0

Terakhir Diperbarui: 17 Januari

2026 Pemelihara: Tim

Arsitektur Siklus Tinjauan: Triwulanan