PRD: Tetapan & Keselamatan - Kawalan Sistem

Kod PRD: KLINIK-Tetapan-PR2026-01-kawalan-sistem-keselamatan **Dicipta:** 13 Januari 2026 **Penulis:** AI Assistant **Dikemaskini:** 13 Januari 2026

---

1. Ringkasan Eksekutif

1.1 Gambaran Keseluruhan

Modul Tetapan & Keselamatan adalah modul teras yang mengurus pengguna sistem, kawalan akses berasaskan peranan (RBAC), audit trail, dan konfigurasi keselamatan untuk memastikan sistem Poliklinik Al-Huda beroperasi dengan selamat dan mematuhi PDPA.

1.2 Metadata

- **Nama Feature**: Sistem Tetapan & Kawalan Keselamatan
- **Modul**: Tetapan & Keselamatan
- **Submodule**: Kawalan Sistem
- **Peranan Sasaran**: Super Admin, Admin
- **Keutamaan**: Kritikal
- **Status**: Perancangan
- **Anggaran Usaha**: Besar (14 minggu)

1.3 Objektif

- Menyediakan pengurusan pengguna yang komprehensif dengan kawalan akses granular
- Melaksanakan Multi-Factor Authentication (MFA) untuk keselamatan tambahan
- Merekod semua aktiviti sistem dalam audit trail untuk pematuhan PDPA
- Mengawal session pengguna dan mencegah akses tidak sah
- Menyediakan konfigurasi sistem yang fleksibel untuk operasi klinik
- Memastikan backup data yang teratur dan boleh dipulihkan

1.4 Skop

Dalam Skop:

- Pengurusan pengguna (CRUD, activate/deactivate, bulk import)
- Role-Based Access Control (RBAC) dengan granular permissions
- Multi-Factor Authentication (TOTP + Email OTP)
- Audit trail komprehensif dengan retention 7 tahun
- Session management (single session, auto-logout, force logout)
- Password policy dan brute force protection
- IP whitelist untuk akses admin
- Konfigurasi sistem (maklumat klinik, waktu operasi)
- Backup & recovery dengan UI dashboard
- Activity monitoring dashboard

Luar Skop:

- Integrasi Single Sign-On (SSO) dengan external providers
- Biometric authentication
- Hardware security keys (YubiKey)
- Penetration testing automation
- Security Information and Event Management (SIEM) integration

---

2. Pernyataan Masalah

2.1 Masalah Semasa

Sistem sedia ada menghadapi isu akses tidak terkawal di mana:

- Pengguna berkongsi akaun kerana tiada kawalan login yang ketat
- Tiada rekod audit untuk mengesan siapa yang melakukan perubahan data
- Password policy tidak dikuatkuasakan menyebabkan kata laluan yang lemah
- Tiada MFA menyebabkan risiko akses tidak sah tinggi
- Konfigurasi sistem tersebar dalam kod dan sukar untuk dikemaskini
- Backup dilakukan secara manual dan tidak konsisten

2.2 Impak Kepada Perniagaan

- **Risiko Keselamatan**: Data pesakit terdedah kepada akses tidak sah
- **Pematuhan PDPA**: Klinik boleh dikenakan penalti kerana tiada audit trail
- **Akauntabiliti**: Sukar mengenal pasti pelaku jika berlaku insiden
- **Operasi**: Downtime yang panjang jika berlaku masalah kerana tiada backup
- **Reputasi**: Kehilangan kepercayaan pesakit jika berlaku kebocoran data

2.3 Hasil Yang Diingini

- Setiap pengguna mempunyai akaun individu dengan akses mengikut peranan
- MFA dikuatkuasakan untuk pengguna kritikal (Admin)
- Semua aktiviti direkod dan boleh diaudit bila-bila masa
- Password policy yang ketat mengurangkan risiko akaun dikompromi
- Konfigurasi sistem boleh dikemaskini tanpa perubahan kod
- Backup automatik dan boleh dipulihkan dengan cepat

---

3. User Stories

3.1 User Stories Utama

- **Sebagai** Super Admin, **saya mahu** menambah pengguna baharu dengan peranan yang sesuai **supaya** setiap staf mempunyai akses mengikut tanggungjawab mereka

- **Sebagai** Super Admin, **saya mahu** menetapkan permissions granular untuk setiap role **supaya** kawalan akses adalah tepat dan mengikut prinsip least privilege

- **Sebagai** Admin, **saya mahu** melihat senarai semua pengguna aktif dan tidak aktif **supaya** saya dapat mengurus akaun staf dengan efisien

- **Sebagai** Admin, **saya mahu** reset password pengguna yang terlupa kata laluan **supaya** mereka boleh akses sistem semula dengan selamat

- **Sebagai** Admin, **saya mahu** menyahaktifkan akaun staf yang telah berhenti kerja **supaya** akses mereka ke sistem ditamatkan serta-merta

- **Sebagai** Pengguna, **saya mahu** mendaftar MFA menggunakan authenticator app **supaya** akaun saya lebih selamat daripada akses tidak sah

- **Sebagai** Admin, **saya mahu** melihat audit log semua aktiviti sistem **supaya** saya dapat menyiasat jika berlaku insiden keselamatan

- **Sebagai** Super Admin, **saya mahu** mengkonfigurasi tetapan klinik seperti waktu operasi **supaya** sistem beroperasi mengikut keperluan semasa

- **Sebagai** Super Admin, **saya mahu** menjalankan backup database secara manual atau berjadual **supaya** data klinik sentiasa selamat dan boleh dipulihkan

- **Sebagai** Admin, **saya mahu** melihat dashboard aktiviti real-time **supaya** saya dapat memantau penggunaan sistem dan mengesan anomali

3.2 Edge Cases & User Stories Sekunder

- **Sebagai** Pengguna, **bila** saya gagal login 5 kali berturut-turut, **saya sepatutnya** melihat mesej akaun dikunci selama 30 minit dan diberikan pilihan untuk reset password
- **Sebagai** Pengguna, **bila** password saya telah tamat tempoh (90 hari), **saya sepatutnya** dipaksa untuk menukar password baharu sebelum boleh meneruskan
- **Sebagai** Admin, **bila** pengguna cuba login dari IP yang tidak dalam whitelist, **saya sepatutnya** menerima alert dan akses tersebut disekat
- **Sebagai** Super Admin, **bila** saya membuang permission daripada role yang sedang digunakan, **saya sepatutnya** melihat amaran tentang pengguna yang akan terjejas
- **Sebagai** Pengguna, **bila** saya kehilangan akses kepada authenticator app, **saya sepatutnya** boleh menggunakan recovery codes yang telah disimpan
- **Sebagai** Admin, **bila** session pengguna telah idle melebihi had masa, **saya sepatutnya** melihat pengguna tersebut auto-logged out dalam senarai active users
- **Sebagai** Super Admin, **bila** backup gagal, **saya sepatutnya** menerima notifikasi email dengan butiran ralat
- **Sebagai** Admin, **bila** pengguna export data pesakit, **saya sepatutnya** melihat rekod tersebut dalam audit log dengan detail lengkap

---

4. Keperluan Fungsian

4.1 Pengurusan Pengguna (FR-100 Series)

- ✓ **FR-101:** Sistem mesti membenarkan Super Admin untuk menambah pengguna baharu dengan maklumat: nama, email, no telefon, role, dan status aktif
- ✓ **FR-102:** Sistem mesti membenarkan Admin untuk melihat senarai semua pengguna dengan fungsi carian dan penapisan mengikut role/status
- ✓ **FR-103:** Sistem mesti membenarkan Super Admin untuk mengemaskini maklumat pengguna termasuk menukar role
- ✓ **FR-104:** Sistem mesti membenarkan Super Admin untuk menyahaktifkan (soft delete) pengguna tanpa membuang rekod
- ✓ **FR-105:** Sistem mesti membenarkan Admin untuk reset password pengguna dan menghantar link reset melalui email
- ✓ **FR-106:** Sistem mesti membenarkan Super Admin untuk import pengguna secara bulk melalui fail CSV/Excel
- ✓ **FR-107:** Sistem mesti membenarkan Admin untuk force logout pengguna yang sedang aktif
- ✓ **FR-108:** Sistem mesti memaparkan last login, last activity, dan device info untuk setiap pengguna

4.2 Role & Permission Management (FR-200 Series)

- ✓ **FR-201:** Sistem mesti menyediakan 6 role default: Super Admin, Admin, Doktor, Jururawat, Kerani, Farmasi
- ✓ **FR-202:** Sistem mesti membenarkan Super Admin untuk menambah role baharu dengan nama dan deskripsi
- ✓ **FR-203:** Sistem mesti menyediakan permissions granular untuk setiap modul (view, create, update, delete, export)
- ✓ **FR-204:** Sistem mesti membenarkan Super Admin untuk assign/unassign permissions kepada role
- ✓ **FR-205:** Sistem mesti membenarkan assign multiple roles kepada seorang pengguna
- ✓ **FR-206:** Sistem mesti memaparkan amaran jika role yang diubah akan menjejaskan pengguna sedia ada
- ✓ **FR-207:** Sistem mesti menyekat pembuangan role yang masih mempunyai pengguna assigned
- ✓ **FR-208:** Sistem mesti menyediakan permission matrix view untuk melihat semua role dan permissions

4.3 Multi-Factor Authentication (FR-300 Series)

- ✅ **FR-301:** Sistem mesti menyokong TOTP-based MFA menggunakan authenticator apps (Google Authenticator, Authy, Microsoft Authenticator)
- ✅ **FR-302:** Sistem mesti menyokong Email OTP sebagai MFA backup method
- ✅ **FR-303:** Sistem mesti menjana 10 recovery codes semasa setup MFA untuk emergency access
- ✅ **FR-304:** Sistem mesti memaksa MFA untuk pengguna dengan role Super Admin dan Admin
- ✅ **FR-305:** Sistem mesti membenarkan pengguna lain untuk opt-in MFA secara sukarela
- ✅ **FR-306:** Sistem mesti membenarkan Admin untuk reset MFA pengguna yang kehilangan akses
- ✅ **FR-307:** Sistem mesti menyimpan trusted devices dan membenarkan skip MFA untuk device yang sama selama 30 hari
- ✅ **FR-308:** Sistem mesti merekod semua percubaan MFA (berjaya/gagal) dalam audit log

4.4 Audit Trail (FR-400 Series)

- ✅ **FR-401:** Sistem mesti merekod semua login dan logout dengan timestamp, IP address, user agent, dan lokasi (jika available)
- ✅ **FR-402:** Sistem mesti merekod semua operasi CRUD pada entiti kritikal (pesakit, EMR, preskripsi, bil, dll)
- ✅ **FR-403:** Sistem mesti merekod perubahan data sensitif dengan nilai sebelum dan selepas (old_value, new_value)
- ✅ **FR-404:** Sistem mesti merekod semua operasi export data dengan detail: pengguna, masa, jenis data, format, bilangan rekod
- ✅ **FR-405:** Sistem mesti menyimpan audit log selama 7 tahun mengikut keperluan PDPA
- ✅ **FR-406:** Sistem mesti membenarkan Admin untuk mencari dan menapis audit log mengikut pengguna, tarikh, jenis aktiviti, dan entiti
- ✅ **FR-407:** Sistem mesti membenarkan export audit log ke format PDF dan CSV untuk keperluan siasatan
- ✅ **FR-408:** Audit log mesti immutable dan tidak boleh diubah atau dipadam oleh sesiapa

4.5 Session Management (FR-500 Series)

- ✅ **FR-501:** Sistem mesti menghadkan setiap pengguna kepada single active session sahaja
- ✅ **FR-502:** Sistem mesti auto-logout pengguna selepas idle timeout (configurable, default 30 minit)
- ✅ **FR-503:** Sistem mesti membenarkan Admin untuk melihat senarai active sessions dengan detail device dan lokasi
- ✅ **FR-504:** Sistem mesti membenarkan Admin untuk force terminate session pengguna
- ✅ **FR-505:** Sistem mesti membenarkan pengguna untuk logout dari semua devices sekaligus
- ✅ **FR-506:** Sistem mesti memaparkan notifikasi jika pengguna login dari device/lokasi baharu
- ✅ **FR-507:** Sistem mesti merekod session history termasuk login time, logout time, dan logout reason

4.6 Password Policy & Security (FR-600 Series)

- ✅ **FR-601:** Sistem mesti menguat kuasa password minimum 12 aksara dengan uppercase, lowercase, nombor, dan simbol
- ✅ **FR-602:** Sistem mesti menyimpan password history (5 passwords terakhir) dan menyekat penggunaan semula
- ✅ **FR-603:** Sistem mesti memaksa password expiry setiap 90 hari
- ✅ **FR-604:** Sistem mesti melaksanakan rate limiting: 5 percubaan gagal = kunci akaun 30 minit
- ✅ **FR-605:** Sistem mesti menyokong IP whitelist untuk akses admin panel
- ✅ **FR-606:** Sistem mesti menyekat login dari IP yang tidak dalam whitelist (untuk admin roles)
- ✅ **FR-607:** Sistem mesti menghantar email alert untuk aktiviti mencurigakan (multiple failed logins, login dari lokasi baharu)
- ✅ **FR-608:** Sistem mesti menyediakan password strength meter semasa create/change password

4.7 System Configuration (FR-700 Series)

- ✅ **FR-701:** Sistem mesti membenarkan konfigurasi maklumat klinik (nama, alamat, no telefon, email, logo)
- ✅ **FR-702:** Sistem mesti membenarkan konfigurasi waktu operasi klinik mengikut hari
- ✅ **FR-703:** Sistem mesti membenarkan konfigurasi notification settings (email, SMS)

- ✅ **FR-704:** Sistem mesti membenarkan konfigurasi backup schedule (harian, mingguan)
- ✅ **FR-705:** Sistem mesti membenarkan konfigurasi session timeout dan password policy parameters
- ✅ **FR-706:** Sistem mesti menyimpan semua configuration changes dalam audit log
- ✅ **FR-707:** Sistem mesti menyediakan feature flags untuk enable/disable modul tertentu

4.8 Backup & Recovery (FR-800 Series)

- ✅ **FR-801:** Sistem mesti membenarkan Super Admin untuk trigger manual backup
- ✅ **FR-802:** Sistem mesti menjalankan scheduled backup mengikut konfigurasi (default: harian 2:00 AM)
- ✅ **FR-803:** Sistem mesti menyimpan backup files dengan encryption (AES-256)
- ✅ **FR-804:** Sistem mesti menyimpan backup di lokasi yang berbeza (local + cloud storage)
- ✅ **FR-805:** Sistem mesti memaparkan backup status dashboard dengan senarai backup files, saiz, dan status
- ✅ **FR-806:** Sistem mesti membenarkan Super Admin untuk restore dari backup tertentu
- ✅ **FR-807:** Sistem mesti menghantar notification jika backup gagal
- ✅ **FR-808:** Sistem mesti mengekalkan backup retention mengikut polisi (default: 30 hari local, 90 hari cloud)

4.9 Activity Dashboard (FR-900 Series)

- ✅ **FR-901:** Sistem mesti memaparkan real-time count pengguna aktif
- ✅ **FR-902:** Sistem mesti memaparkan login history untuk 24 jam terakhir dalam bentuk chart
- ✅ **FR-903:** Sistem mesti memaparkan security alerts (failed logins, suspicious activities)
- ✅ **FR-904:** Sistem mesti memaparkan system health metrics (CPU, memory, disk usage)
- ✅ **FR-905:** Sistem mesti memaparkan recent audit log entries
- ✅ **FR-906:** Sistem mesti membenarkan drill-down ke detail untuk setiap metric

4.2 Kebenaran & Kawalan Akses

Peranan Diperlukan:

- Super Admin: Akses penuh ke semua fungsi
- Admin: Akses terhad (tidak boleh ubah role Super Admin, tidak boleh delete audit log)

Kebenaran Diperlukan:

Permission	Super Admin	Admin
`users.view`	✓	✓
`users.create`	✓	✓
`users.update`	✓	✓
`users.delete`	✓	✗
`users.import`	✓	✗
`roles.view`	✓	✓
`roles.create`	✓	✗
`roles.update`	✓	✗
`roles.delete`	✓	✗
`permissions.manage`	✓	✗
`audit.view`	✓	✓
`audit.export`	✓	✓

`settings.view`	✓	✓
`settings.update`	✓	✗
`backup.view`	✓	✓
`backup.create`	✓	✗
`backup.restore`	✓	✗
`mfa.reset`	✓	✓
`sessions.view`	✓	✓
`sessions.terminate`	✓	✓

Authorization Logic:

- Super Admin tidak boleh deactivate diri sendiri
- Super Admin tidak boleh remove role Super Admin dari diri sendiri
- Mesti ada sekurang-kurangnya seorang Super Admin aktif dalam sistem
- Admin tidak boleh mengubah atau melihat maklumat Super Admin

4.3 Validasi Data

Field Wajib - Pengguna:

- `nama` : Required, string, max 255
- `email` : Required, email format, unique
- `password` : Required (on create), min 12 chars, complexity rules
- `role_id` : Required, exists in roles table

Field Wajib - Role:

- `nama` : Required, string, max 100, unique
- `kod` : Required, string, max 50, unique, uppercase
- `deskripsi` : Optional, string, max 500

Peraturan Validasi:

- Email mesti unik dalam sistem
- Password mesti mengikut complexity rules (12 chars, mixed case, number, symbol)
- Password tidak boleh sama dengan 5 password sebelumnya
- Role kod mesti uppercase dan alphanumeric sahaja
- IP whitelist mesti valid IP address atau CIDR notation

Peraturan Perniagaan:

- Tidak boleh delete pengguna yang mempunyai rekod audit (soft delete sahaja)
- Tidak boleh delete role yang masih mempunyai pengguna assigned
- Tidak boleh deactivate Super Admin terakhir
- MFA wajib untuk Super Admin dan Admin
- Password expiry dikecualikan untuk service accounts

4.4 Audit Trail & PDPA Compliance

- ✓ **Adakah feature ini perlu audit trail?** Ya - Kritikal
- **Field Audit**: created_by, updated_by, deleted_at, created_at, updated_at
- **Data Consent**: Tidak berkaitan (modul ini bukan data pesakit)
- **Data Retention**: Audit log disimpan selama 7 tahun, kemudian di-archive

Audit Events:

Event Category	Events
Authentication	login, logout, login_failed, mfa_setup, mfa_verified, mfa_failed, password_changed, password_reset
User Management	user_created, user_updated, user_deactivated, user_activated, user_deleted
Role Management	role_created, role_updated, role_deleted, permission_assigned, permission_revoked
Session	session_started, session_ended, session_terminated, session_timeout
Security	account_locked, account_unlocked, ip_blocked, suspicious_activity
System	settings_updated, backup_created, backup_restored, backup_failed
Data Access	data_exported, report_generated, bulk_import

---

5. Keperluan Teknikal

5.1 Teknologi Stack

- **Framework**: Laravel 12
- **Frontend**: Blade Templates + Bootstrap 5 + CoreUI
- **Database**: MySQL 8.0
- **Authentication**: Laravel Breeze + Custom MFA
- **Authorization**: Spatie Laravel-Permission
- **TOTP**: pragmarx/google2fa-laravel
- **Queue**: Laravel Queue (database driver)
- **Cache**: Redis (untuk rate limiting dan session)
- **Encryption**: Laravel Encryption (AES-256-CBC)
- **Backup**: spatie/laravel-backup

5.2 Arsitektur Aplikasi

```
Route Attributes (dalam Controller)
   ↓
Controller (HTTP Layer)
   ↓
FormRequest (Validation Layer)
   ↓
Service Layer (Business Logic)
   ↓
Repository Layer (Data Access)
   ↓
Model (Eloquent ORM)
   ↓
Database
```

5.3 Struktur Modul

```
app/
├── Http/
│   ├── Controllers/
│   │   └── Admin/
│   │       ├── UserController.php
│   │       ├── RoleController.php
│   │       ├── PermissionController.php
│   │       ├── AuditLogController.php
│   │       ├── SettingsController.php
│   │       ├── BackupController.php
│   │       ├── SessionController.php
│   │       └── SecurityDashboardController.php
│   ├── Requests/
│   │   ├── StoreUserRequest.php
│   │   ├── UpdateUserRequest.php
│   │   ├── StoreRoleRequest.php
│   │   ├── UpdateRoleRequest.php
│   │   ├── UpdateSettingsRequest.php
│   │   └── VerifyMfaRequest.php
│   └── Middleware/
│       ├── EnforceMfa.php
│       ├── CheckIpWhitelist.php
│       ├── SingleSession.php
│       └── AuditActivity.php
├── Services/
│   ├── UserService.php
│   ├── RoleService.php
│   ├── MfaService.php
│   ├── AuditLogService.php
│   ├── SessionService.php
│   ├── BackupService.php
│   └── SecurityService.php
├── Repositories/
│   ├── UserRepository.php
│   ├── RoleRepository.php
│   ├── AuditLogRepository.php
│   └── SettingsRepository.php
├── Models/
│   ├── User.php (extend)
│   ├── Role.php (Spatie)
│   ├── Permission.php (Spatie)
│   ├── AuditLog.php
│   ├── UserMfa.php
│   ├── UserSession.php
│   ├── LoginAttempt.php
│   ├── PasswordHistory.php
│   ├── IpWhitelist.php
│   ├── SystemSetting.php
│   ├── BackupLog.php
│   └── TrustedDevice.php
├── Observers/
│   └── AuditableObserver.php
├── Listeners/
│   ├── LogSuccessfulLogin.php
│   ├── LogFailedLogin.php
│   ├── LogLogout.php
│   └── SendSecurityAlert.php
├── Notifications/
│   ├── MfaSetupNotification.php
│   ├── PasswordResetNotification.php
│   ├── AccountLockedNotification.php
│   ├── NewDeviceLoginNotification.php
│   └── BackupFailedNotification.php
└── Console/
```

```
    └── Commands/
        ├── PurgeOldAuditLogs.php
        ├── RunScheduledBackup.php
        └── CleanExpiredSessions.php

config/
├── tetapan.php
├── security.php
└── audit.php

resources/
└── views/
    └── admin/
        ├── users/
        │   ├── index.blade.php
        │   ├── create.blade.php
        │   ├── edit.blade.php
        │   └── show.blade.php
        ├── roles/
        │   ├── index.blade.php
        │   ├── create.blade.php
        │   ├── edit.blade.php
        │   └── permissions.blade.php
        ├── audit-logs/
        │   ├── index.blade.php
        │   └── show.blade.php
        ├── settings/
        │   ├── index.blade.php
        │   ├── clinic.blade.php
        │   ├── security.blade.php
        │   └── notifications.blade.php
        ├── backup/
        │   ├── index.blade.php
        │   └── restore.blade.php
        ├── sessions/
        │   └── index.blade.php
        ├── mfa/
        │   ├── setup.blade.php
        │   ├── verify.blade.php
        │   └── recovery.blade.php
        └── security-dashboard/
            └── index.blade.php
```

5.4 Database Schema

Jadual: `users` (Extend Laravel Default)

Column	Type	Description
`id`	bigint UNSIGNED PK	Primary key
`nama`	varchar(255) NOT NULL	Nama penuh pengguna
`email`	varchar(255) UNIQUE NOT NULL	Email untuk login
`email_verified_at`	timestamp NULL	Timestamp email verified
`password`	varchar(255) NOT NULL	Hashed password

`no_telefon`	varchar(20) NULL	Nombor telefon
`jawatan`	varchar(100) NULL	Jawatan/designation
`gambar`	varchar(255) NULL	Path to profile picture
`is_active`	boolean DEFAULT true	Status aktif
`mfa_enabled`	boolean DEFAULT false	MFA enabled flag
`mfa_enforced`	boolean DEFAULT false	MFA wajib untuk user ini
`password_changed_at`	timestamp NULL	Last password change
`last_login_at`	timestamp NULL	Last successful login
`last_login_ip`	varchar(45) NULL	Last login IP address
`last_activity_at`	timestamp NULL	Last activity timestamp
`failed_login_attempts`	int DEFAULT 0	Failed login counter
`locked_until`	timestamp NULL	Account locked until
`created_by`	bigint UNSIGNED NULL	FK → users.id
`updated_by`	bigint UNSIGNED NULL	FK → users.id
`remember_token`	varchar(100) NULL	Remember me token
`created_at`	timestamp	Created timestamp
`updated_at`	timestamp	Updated timestamp
`deleted_at`	timestamp NULL	Soft delete

Indexes:

- `idx_users_email` on `email`
- `idx_users_is_active` on `is_active`
- `idx_users_last_login` on `last_login_at`

Jadual: `user_mfa`

Column	Type	Description
`id`	bigint UNSIGNED PK	Primary key
`user_id`	bigint UNSIGNED NOT NULL	FK → users.id
`secret_key`	varchar(255) NOT NULL	Encrypted TOTP secret
`recovery_codes`	text NOT NULL	Encrypted JSON array of recovery codes
`enabled_at`	timestamp NULL	When MFA was enabled
`last_used_at`	timestamp NULL	Last MFA verification
`created_at`	timestamp	Created timestamp
`updated_at`	timestamp	Updated timestamp

Indexes:

- `idx_user_mfa_user` on `user_id`

Foreign Keys:

- `fk_user_mfa_user` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`) ON DELETE CASCADE

Jadual: `trusted_devices`

Column	Type	Description
`id`	bigint UNSIGNED PK	Primary key
`user_id`	bigint UNSIGNED NOT NULL	FK → users.id
`device_hash`	varchar(64) NOT NULL	Hash of device fingerprint
`device_name`	varchar(255) NULL	Browser/device name
`ip_address`	varchar(45) NOT NULL	IP when trusted
`user_agent`	text NULL	Full user agent
`trusted_until`	timestamp NOT NULL	Trust expiry (30 days)
`created_at`	timestamp	Created timestamp

Indexes:

- `idx_trusted_devices_user` on `user_id`
- `idx_trusted_devices_hash` on `device_hash`

Foreign Keys:

- `fk_trusted_devices_user` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`) ON DELETE CASCADE

Jadual: `password_histories`

Column	Type	Description
`id`	bigint UNSIGNED PK	Primary key
`user_id`	bigint UNSIGNED NOT NULL	FK → users.id
`password`	varchar(255) NOT NULL	Hashed old password
`created_at`	timestamp	When password was set

Indexes:

- `idx_password_history_user` on `user_id`

Foreign Keys:

- `fk_password_history_user` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`) ON DELETE CASCADE

Jadual: `login_attempts`

Column	Type	Description
`id`	bigint UNSIGNED PK	Primary key
`email`	varchar(255) NOT NULL	Attempted email
`user_id`	bigint UNSIGNED NULL	FK → users.id (if exists)

`ip_address`	varchar(45) NOT NULL	IP address
`user_agent`	text NULL	User agent string
`is_successful`	boolean NOT NULL	Success/failure
`failure_reason`	varchar(100) NULL	Reason for failure
`mfa_required`	boolean DEFAULT false	MFA was required
`mfa_passed`	boolean NULL	MFA verification result
`created_at`	timestamp	Attempt timestamp

Indexes:

- `idx_login_attempts_email` on `email`
- `idx_login_attempts_ip` on `ip_address`
- `idx_login_attempts_created` on `created_at`

Jadual: `user_sessions`

Column	Type	Description
`id`	varchar(255) PK	Session ID
`user_id`	bigint UNSIGNED NOT NULL	FK → users.id
`ip_address`	varchar(45) NOT NULL	IP address
`user_agent`	text NULL	User agent
`device_type`	varchar(50) NULL	desktop/mobile/tablet
`browser`	varchar(100) NULL	Browser name
`os`	varchar(100) NULL	Operating system
`location`	varchar(255) NULL	Geo location (city, country)
`payload`	longtext NOT NULL	Session payload
`last_activity`	int NOT NULL	Last activity timestamp
`login_at`	timestamp NOT NULL	Login timestamp
`logout_at`	timestamp NULL	Logout timestamp
`logout_reason`	varchar(50) NULL	manual/timeout/forced/expired

Indexes:

- `idx_user_sessions_user` on `user_id`
- `idx_user_sessions_activity` on `last_activity`

Jadual: `audit_logs`

Column	Type	Description
`id`	bigint UNSIGNED PK	Primary key
`user_id`	bigint UNSIGNED NULL	FK → users.id (NULL for system)

`user_nama`	varchar(255) NULL	Snapshot of user name
`user_email`	varchar(255) NULL	Snapshot of user email
`event_type`	varchar(100) NOT NULL	Event type (login, create, update, etc.)
`event_category`	varchar(50) NOT NULL	Category (auth, user, data, system)
`auditable_type`	varchar(255) NULL	Model class name
`auditable_id`	bigint UNSIGNED NULL	Model ID
`description`	text NOT NULL	Human-readable description
`old_values`	json NULL	Values before change
`new_values`	json NULL	Values after change
`ip_address`	varchar(45) NULL	IP address
`user_agent`	text NULL	User agent
`url`	varchar(500) NULL	Request URL
`method`	varchar(10) NULL	HTTP method
`metadata`	json NULL	Additional metadata
`created_at`	timestamp	Event timestamp

Indexes:

- `idx_audit_logs_user` on `user_id`
- `idx_audit_logs_type` on `event_type`
- `idx_audit_logs_category` on `event_category`
- `idx_audit_logs_auditable` on `auditable_type`, `auditable_id`
- `idx_audit_logs_created` on `created_at`

Partitioning:

- Partition by RANGE on `created_at` (monthly partitions)

Jadual: `ip_whitelists`

Column	Type	Description
`id`	bigint UNSIGNED PK	Primary key
`ip_address`	varchar(45) NOT NULL	IP address or CIDR
`description`	varchar(255) NULL	Description/label
`is_active`	boolean DEFAULT true	Active status
`created_by`	bigint UNSIGNED NULL	FK → users.id
`created_at`	timestamp	Created timestamp
`updated_at`	timestamp	Updated timestamp

Indexes:

- `idx_ip_whitelist_ip` on `ip_address`

- `idx_ip_whitelist_active` on `is_active`

Jadual: `system_settings`

Column	Type	Description
`id`	bigint UNSIGNED PK	Primary key
`group`	varchar(50) NOT NULL	Setting group (clinic, security, notification, backup)
`key`	varchar(100) NOT NULL	Setting key
`value`	text NULL	Setting value
`type`	varchar(20) DEFAULT 'string'	Data type (string, integer, boolean, json)
`description`	varchar(255) NULL	Description
`is_encrypted`	boolean DEFAULT false	Value is encrypted
`updated_by`	bigint UNSIGNED NULL	FK → users.id
`created_at`	timestamp	Created timestamp
`updated_at`	timestamp	Updated timestamp

Indexes:

- `idx_system_settings_group_key` UNIQUE on `group` , `key`

Jadual: `backup_logs`

Column	Type	Description
`id`	bigint UNSIGNED PK	Primary key
`filename`	varchar(255) NOT NULL	Backup filename
`disk`	varchar(50) NOT NULL	Storage disk (local, s3)
`path`	varchar(500) NOT NULL	Full path to backup
`size_bytes`	bigint NOT NULL	File size in bytes
`type`	varchar(20) NOT NULL	manual/scheduled
`status`	varchar(20) NOT NULL	pending/running/completed/failed
`started_at`	timestamp NULL	Backup start time
`completed_at`	timestamp NULL	Backup completion time
`error_message`	text NULL	Error message if failed
`created_by`	bigint UNSIGNED NULL	FK → users.id
`created_at`	timestamp	Created timestamp

Indexes:

- `idx_backup_logs_status` on `status`
- `idx_backup_logs_created` on `created_at`

5.5 Model Eloquent

Model: `AuditLog`

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class AuditLog extends Model
{
    public $timestamps = false;

    protected $table = 'audit_logs';

    protected $fillable = [
        'user_id',
        'user_nama',
        'user_email',
        'event_type',
        'event_category',
        'auditable_type',
        'auditable_id',
        'description',
        'old_values',
        'new_values',
        'ip_address',
        'user_agent',
        'url',
        'method',
        'metadata',
        'created_at',
    ];

    protected $casts = [
        'old_values' => 'array',
        'new_values' => 'array',
        'metadata' => 'array',
        'created_at' => 'datetime',
    ];

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

    public function auditable()
    {
        return $this->morphTo();
    }

    // Scopes
    public function scopeByUser($query, $userId)
    {
        return $query->where('user_id', $userId);
    }
    public function scopeByCategory($query, $category)
    {
        return $query->where('event_category', $category);
    }

    public function scopeByEventType($query, $type)
    {
        return $query->where('event_type', $type);
    }

    public function scopeBetweenDates($query, $startDate, $endDate)
    {
        return $query->whereBetween('created_at', [$startDate, $endDate]);
    }

    // Boot - Disable updates and deletes
    protected static function boot()
    {
        parent::boot();

        static::updating(function ($model) {
            return false; // Prevent updates
        });

        static::deleting(function ($model) {
            return false; // Prevent deletes
        });
    }
}
```

Model: `UserMfa`

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\Crypt;

class UserMfa extends Model
{
    protected $table = 'user_mfa';

    protected $fillable = [
        'user_id',
        'secret_key',
        'recovery_codes',
        'enabled_at',
        'last_used_at',
    ];

    protected $casts = [
        'enabled_at' => 'datetime',
        'last_used_at' => 'datetime',
```

```php
    ];

    protected $hidden = [
        'secret_key',
        'recovery_codes',
    ];

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

    // Accessors & Mutators
    public function setSecretKeyAttribute($value)
    {
        $this->attributes['secret_key'] = Crypt::encryptString($value);
    }

    public function getSecretKeyAttribute($value)
    {
        return Crypt::decryptString($value);
    }

    public function setRecoveryCodesAttribute($value)
    {
        $this->attributes['recovery_codes'] = Crypt::encryptString(json_encode($value));
    }

    public function getRecoveryCodesAttribute($value)
    {
        return json_decode(Crypt::decryptString($value), true);
    }

    // Methods
    public function useRecoveryCode($code)
    {
        $codes = $this->recovery_codes;
        $index = array_search($code, $codes);

        if ($index !== false) {
            unset($codes[$index]);
            $this->recovery_codes = array_values($codes);
            $this->save();
            return true;
        }

        return false;
    }
}
```

Model: `SystemSetting`

```php
<?php
```

```php
namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\Cache;
use Illuminate\Support\Facades\Crypt;

class SystemSetting extends Model
{
    protected $table = 'system_settings';

    protected $fillable = [
        'group',
        'key',
        'value',
        'type',
        'description',
        'is_encrypted',
        'updated_by',
    ];

    protected $casts = [
        'is_encrypted' => 'boolean',
    ];

    // Relationships
    public function updater()
    {
        return $this->belongsTo(User::class, 'updated_by');
    }

    // Accessors
    public function getValueAttribute($value)
    {
        if ($this->is_encrypted && $value) {
            $value = Crypt::decryptString($value);
        }

        return match($this->type) {
            'integer' => (int) $value,
            'boolean' => filter_var($value, FILTER_VALIDATE_BOOLEAN),
            'json' => json_decode($value, true),
            default => $value,
        };
    }

    // Mutators
    public function setValueAttribute($value)
    {
        if ($this->type === 'json' && is_array($value)) {
            $value = json_encode($value);
        }

        if ($this->is_encrypted) {
            $value = Crypt::encryptString($value);
        }
        $this->attributes['value'] = $value;
    }

    // Static Methods
    public static function get($group, $key, $default = null)
    {
        $cacheKey = "settings.{$group}.{$key}";

        return Cache::remember($cacheKey, 3600, function () use ($group, $key, $default) {
            $setting = static::where('group', $group)
                ->where('key', $key)
                ->first();

            return $setting ? $setting->value : $default;
        });
    }

    public static function set($group, $key, $value, $userId = null)
    {
        $setting = static::updateOrCreate(
            ['group' => $group, 'key' => $key],
            ['value' => $value, 'updated_by' => $userId]
        );

        Cache::forget("settings.{$group}.{$key}");

        return $setting;
    }

    public static function getGroup($group)
    {
        return Cache::remember("settings.{$group}", 3600, function () use ($group) {
            return static::where('group', $group)
                ->pluck('value', 'key')
                ->toArray();
        });
    }
}
```

5.6 Configuration Files

File: `config/security.php`

```
<?php

return [
    /*
    |--------------------------------------------------------------------------
    | Password Policy
    |--------------------------------------------------------------------------
    */
    'password' => [
        'min_length' => 12,
        'require_uppercase' => true,
        'require_lowercase' => true,
        'require_numbers' => true,
        'require_symbols' => true,
        'history_count' => 5,
        'expiry_days' => 90,
        'expiry_warning_days' => 14,
    ],

    /*
    |--------------------------------------------------------------------------
    | Account Lockout Policy
    |--------------------------------------------------------------------------
    */
    'lockout' => [
        'max_attempts' => 5,
        'lockout_minutes' => 30,
        'decay_minutes' => 60,
    ],

    /*
    |--------------------------------------------------------------------------
    | Session Configuration
    |--------------------------------------------------------------------------
    */
    'session' => [
        'single_session' => true,
        'idle_timeout_minutes' => 30,
        'absolute_timeout_hours' => 12,
        'extend_on_activity' => true,
    ],

    /*
    |--------------------------------------------------------------------------
    | MFA Configuration
    |--------------------------------------------------------------------------
    */
    'mfa' => [
        'enabled' => true,
        'enforced_roles' => ['super-admin', 'admin'],
        'totp_enabled' => true,
        'email_otp_enabled' => true,
        'recovery_codes_count' => 10,
        'trusted_device_days' => 30,
        'totp_window' => 1, // Allow 1 period before/after
    ],

    /*
    |--------------------------------------------------------------------------
    | IP Whitelist
    |--------------------------------------------------------------------------
    */
    'ip_whitelist' => [
        'enabled' => true,
        'enforced_roles' => ['super-admin', 'admin'],
        'bypass_for_local' => true,
        'local_ips' => ['127.0.0.1', '::1'],
    ],
    /*
    |--------------------------------------------------------------------------
    | Security Alerts
    |--------------------------------------------------------------------------
    */
    'alerts' => [
        'failed_login_threshold' => 3,
        'new_device_login' => true,
        'new_location_login' => true,
        'account_lockout' => true,
        'password_change' => true,
        'mfa_disabled' => true,
    ],
];
```

File: `config/audit.php`

```php
<?php

return [
    /*
    |--------------------------------------------------------------------------
    | Audit Log Configuration
    |--------------------------------------------------------------------------
    */
    'enabled' => true,

    /*
    |--------------------------------------------------------------------------
    | Retention Period
    |--------------------------------------------------------------------------
    */
    'retention_years' => 7,

    /*
    |--------------------------------------------------------------------------
    | Events to Log
    |--------------------------------------------------------------------------
    */
    'events' => [
        'authentication' => [
            'login',
            'logout',
            'login_failed',
            'mfa_setup',
            'mfa_verified',
            'mfa_failed',
            'password_changed',
            'password_reset',
        ],
        'user_management' => [
            'user_created',
            'user_updated',
            'user_deactivated',
            'user_activated',
            'user_deleted',
        ],
        'role_management' => [
            'role_created',
            'role_updated',
            'role_deleted',
            'permission_assigned',
            'permission_revoked',
        ],
        'session' => [
            'session_started',
            'session_ended',
            'session_terminated',
            'session_timeout',
        ],
        'security' => [
            'account_locked',
            'account_unlocked',
            'ip_blocked',
            'suspicious_activity',
        ],
        'system' => [
            'settings_updated',
            'backup_created',
            'backup_restored',
            'backup_failed',
        ],
        'data_access' => [
            'data_exported',
            'report_generated',
            'bulk_import',
        ],
    ],

    /*
    |--------------------------------------------------------------------------
    | Sensitive Fields (akan di-mask dalam log)
    |--------------------------------------------------------------------------
    */
    'sensitive_fields' => [
        'password',
        'password_confirmation',
        'current_password',
        'secret_key',
        'recovery_codes',
        'api_key',
        'api_secret',
    ],

    /*
    |--------------------------------------------------------------------------
    | Models to Audit
    |--------------------------------------------------------------------------
    */
    'auditable_models' => [
        \App\Models\User::class,
        \App\Models\Pesakit::class,
```

```php
        \App\Models\RekodPerubatan::class,
        \App\Models\Preskripsi::class,
        \App\Models\Invoice::class,
        \App\Models\Temujanji::class,
    ],

    /*
    |--------------------------------------------------------------------------
    | Export Configuration
    |--------------------------------------------------------------------------
    */
    'export' => [
        'formats' => ['pdf', 'csv'],
        'max_records' => 10000,
        'chunk_size' => 1000,
    ],
];
```

File: `config/tetapan.php`

```php
<?php

return [
    /*
    |--------------------------------------------------------------------------
    | Default Clinic Settings
    |--------------------------------------------------------------------------
    */
    'clinic' => [
        'nama' => 'Poliklinik Al-Huda',
        'alamat' => '',
        'poskod' => '',
        'bandar' => '',
        'negeri' => '',
        'no_telefon' => '',
        'no_faks' => '',
        'email' => '',
        'website' => '',
        'no_pendaftaran_moh' => '',
        'no_lesen' => '',
    ],

    /*
    |--------------------------------------------------------------------------
    | Operating Hours
    |--------------------------------------------------------------------------
    */
    'waktu_operasi' => [
        'isnin' => ['buka' => '08:00', 'tutup' => '17:00', 'aktif' => true],
        'selasa' => ['buka' => '08:00', 'tutup' => '17:00', 'aktif' => true],
        'rabu' => ['buka' => '08:00', 'tutup' => '17:00', 'aktif' => true],
        'khamis' => ['buka' => '08:00', 'tutup' => '17:00', 'aktif' => true],
        'jumaat' => ['buka' => '08:00', 'tutup' => '17:00', 'aktif' => true],
        'sabtu' => ['buka' => '08:00', 'tutup' => '13:00', 'aktif' => true],
        'ahad' => ['buka' => null, 'tutup' => null, 'aktif' => false],
    ],
    /*
    |--------------------------------------------------------------------------
    | Backup Configuration
    |--------------------------------------------------------------------------
    */
    'backup' => [
        'schedule' => 'daily', // daily, weekly
        'time' => '02:00',
        'retention_days_local' => 30,
        'retention_days_cloud' => 90,
        'notify_on_failure' => true,
        'notify_emails' => [],
    ],

    /*
    |--------------------------------------------------------------------------
    | Default Roles
    |--------------------------------------------------------------------------
    */
    'default_roles' => [
        [
            'name' => 'super-admin',
            'display_name' => 'Super Admin',
            'description' => 'Akses penuh kepada semua fungsi sistem',
            'is_system' => true,
        ],
        [
            'name' => 'admin',
            'display_name' => 'Admin',
            'description' => 'Pengurusan sistem dan pengguna',
            'is_system' => true,
        ],
        [
            'name' => 'doktor',
            'display_name' => 'Doktor',
            'description' => 'Akses kepada rekod perubatan dan preskripsi',
            'is_system' => false,
        ],
        [
            'name' => 'jururawat',
            'display_name' => 'Jururawat',
            'description' => 'Akses kepada vital signs dan assist doktor',
            'is_system' => false,
        ],
        [
            'name' => 'kerani',
            'display_name' => 'Kerani',
            'description' => 'Pendaftaran, temujanji, dan billing',
            'is_system' => false,
        ],
        [
            'name' => 'farmasi',
            'display_name' => 'Farmasi',
            'description' => 'Pengurusan ubat dan dispensing',
            'is_system' => false,
        ],
    ],

    /*
    |--------------------------------------------------------------------------
    | Module Permissions
    |--------------------------------------------------------------------------
    */
    'module_permissions' => [
        'pendaftaran' => ['view', 'create', 'update', 'delete', 'export'],
        'temujanji' => ['view', 'create', 'update', 'delete', 'export'],
        'emr' => ['view', 'create', 'update', 'delete', 'export', 'print'],
        'farmasi' => ['view', 'create', 'update', 'delete', 'export', 'dispense'],
        'billing' => ['view', 'create', 'update', 'delete', 'export', 'refund', 'void'],
        'panel' => ['view', 'create', 'update', 'delete', 'export', 'verify', 'claim'],
        'queue' => ['view', 'create', 'update', 'delete', 'call', 'transfer'],
        'laporan' => ['view', 'export', 'schedule'],
        'tetapan' => ['view', 'update'],
        'pengguna' => ['view', 'create', 'update', 'delete', 'import'],
        'roles' => ['view', 'create', 'update', 'delete', 'assign'],
        'audit' => ['view', 'export'],
        'backup' => ['view', 'create', 'restore'],
    ],
];
```

5.7 Routes (Route Attributes)

File: `app/Http/Controllers/Admin/UserController.php`

```php
<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;
use App\Models\User;
use App\Http\Requests\StoreUserRequest;
use App\Http\Requests\UpdateUserRequest;
use App\Services\UserService;
use App\Traits\HandlesApiResponses;
use Illuminate\Support\Facades\Log;
use Spatie\RouteAttributes\Attributes\Get;
use Spatie\RouteAttributes\Attributes\Post;
use Spatie\RouteAttributes\Attributes\Patch;
use Spatie\RouteAttributes\Attributes\Delete;
use Spatie\RouteAttributes\Attributes\Prefix;
use Spatie\RouteAttributes\Attributes\Middleware;

#[Prefix('admin/users')]
#[Middleware(['web', 'auth', 'mfa.verified'])]
class UserController extends Controller
{
    use HandlesApiResponses;

    protected UserService $service;

    public function __construct(UserService $service)
    {
        $this->service = $service;
    }

    #[Get('/', name: 'admin.users.index')]
    public function index()
    {
        $this->authorize('viewAny', User::class);
        $users = $this->service->getPaginated(request()->all());
        return view('admin.users.index', compact('users'));
    }

    #[Get('/create', name: 'admin.users.create')]
    public function create()
    {
        $this->authorize('create', User::class);
        $roles = $this->service->getAvailableRoles();
        return view('admin.users.create', compact('roles'));
    }

    #[Post('/', name: 'admin.users.store')]
    public function store(StoreUserRequest $request)
    {
        try {
            $this->authorize('create', User::class);
            $this->service->create($request->validated());
            return $this->successRedirect('admin.users.index', 'Pengguna berjaya ditambah');
        } catch (\Exception $e) {
            Log::error('User creation failed', ['error' => $e->getMessage()]);
            return $this->errorRedirect('Gagal menambah pengguna');
        }
    }

    #[Get('/{user}', name: 'admin.users.show')]
    public function show(User $user)
    {
        $this->authorize('view', $user);
        $activityLog = $this->service->getUserActivity($user->id);
        return view('admin.users.show', compact('user', 'activityLog'));
    }

    #[Get('/{user}/edit', name: 'admin.users.edit')]
    public function edit(User $user)
    {
        $this->authorize('update', $user);
        $roles = $this->service->getAvailableRoles();
        return view('admin.users.edit', compact('user', 'roles'));
    }

    #[Patch('/{user}', name: 'admin.users.update')]
    public function update(UpdateUserRequest $request, User $user)
    {
        try {
            $this->authorize('update', $user);
            $this->service->update($user->id, $request->validated());
            return $this->successRedirect('admin.users.index', 'Pengguna berjaya dikemaskini');
        } catch (\Exception $e) {
            Log::error('User update failed', ['id' => $user->id, 'error' => $e->getMessage()]);
            return $this->errorRedirect('Gagal mengemaskini pengguna');
        }
    }

    #[Delete('/{user}', name: 'admin.users.destroy')]
    public function destroy(User $user)
    {
        try {
            $this->authorize('delete', $user);
            $this->service->deactivate($user->id);
            return $this->successRedirect('admin.users.index', 'Pengguna berjaya dinyahaktifkan');
        } catch (\Exception $e) {
            Log::error('User deactivation failed', ['id' => $user->id, 'error' => $e->getMessage()]);
            return $this->errorRedirect('Gagal menyahaktifkan pengguna');
        }
    }

    #[Post('/{user}/reset-password', name: 'admin.users.reset-password')]
    public function resetPassword(User $user)
    {
        try {
            $this->authorize('update', $user);
            $this->service->sendPasswordReset($user->id);
            return $this->successRedirect('admin.users.show', 'Link reset password telah dihantar',
['user' => $user->id]);
        } catch (\Exception $e) {
            Log::error('Password reset failed', ['id' => $user->id, 'error' => $e->getMessage()]);
            return $this->errorRedirect('Gagal menghantar link reset password');
        }
    }

    #[Post('/{user}/toggle-status', name: 'admin.users.toggle-status')]
    public function toggleStatus(User $user)
    {
        try {
            $this->authorize('update', $user);
            $this->service->toggleStatus($user->id);
            $status = $user->fresh()->is_active ? 'diaktifkan' : 'dinyahaktifkan';
            return $this->successRedirect('admin.users.index', "Pengguna berjaya {$status}");
        } catch (\Exception $e) {
            Log::error('User status toggle failed', ['id' => $user->id, 'error' => $e->getMessage()]);
            return $this->errorRedirect('Gagal menukar status pengguna');
        }
    }

    #[Post('/{user}/reset-mfa', name: 'admin.users.reset-mfa')]
    public function resetMfa(User $user)
    {
        try {
            $this->authorize('update', $user);
            $this->service->resetMfa($user->id);
            return $this->successRedirect('admin.users.show', 'MFA berjaya di-reset', ['user' =>
$user->id]);
        } catch (\Exception $e) {
            Log::error('MFA reset failed', ['id' => $user->id, 'error' => $e->getMessage()]);
            return $this->errorRedirect('Gagal reset MFA');
        }
    }

    #[Post('/{user}/force-logout', name: 'admin.users.force-logout')]
    public function forceLogout(User $user)
    {
        try {
            $this->authorize('update', $user);
            $this->service->forceLogout($user->id);
            return $this->successRedirect('admin.users.show', 'Pengguna berjaya di-logout', ['user' =>
$user->id]);
        } catch (\Exception $e) {
            Log::error('Force logout failed', ['id' => $user->id, 'error' => $e->getMessage()]);
            return $this->errorRedirect('Gagal logout pengguna');
        }
    }

    #[Get('/import', name: 'admin.users.import')]
    public function importForm()
    {
        $this->authorize('create', User::class);
        return view('admin.users.import');
    }

    #[Post('/import', name: 'admin.users.import.process')]
    public function import()
    {
        try {
            $this->authorize('create', User::class);
            request()->validate(['file' => 'required|file|mimes:csv,xlsx|max:5120']);
            $result = $this->service->bulkImport(request()->file('file'));
            return $this->successRedirect('admin.users.index', "Berjaya import {$result['success']}
pengguna");
        } catch (\Exception $e) {
            Log::error('User import failed', ['error' => $e->getMessage()]);
            return $this->errorRedirect('Gagal import pengguna');
        }
    }
}
```

Route Summary:

Method	URI	Name	Middleware
GET	`/admin/users`	admin.users.index	web, auth, mfa.verified
GET	`/admin/users/create`	admin.users.create	web, auth, mfa.verified
POST	`/admin/users`	admin.users.store	web, auth, mfa.verified
GET	`/admin/users/{user}`	admin.users.show	web, auth, mfa.verified
GET	`/admin/users/{user}/edit`	admin.users.edit	web, auth, mfa.verified
PATCH	`/admin/users/{user}`	admin.users.update	web, auth, mfa.verified

DELETE	`/admin/users/{user}`	admin.users.destroy	web, auth, mfa.verified
POST	`/admin/users/{user}/reset-password`	admin.users.reset-password	web, auth, mfa.verified
POST	`/admin/users/{user}/toggle-status`	admin.users.toggle-status	web, auth, mfa.verified
POST	`/admin/users/{user}/reset-mfa`	admin.users.reset-mfa	web, auth, mfa.verified
POST	`/admin/users/{user}/force-logout`	admin.users.force-logout	web, auth, mfa.verified
GET	`/admin/users/import`	admin.users.import	web, auth, mfa.verified
POST	`/admin/users/import`	admin.users.import.process	web, auth, mfa.verified
GET	`/admin/roles`	admin.roles.index	web, auth, mfa.verified
GET	`/admin/roles/create`	admin.roles.create	web, auth, mfa.verified
POST	`/admin/roles`	admin.roles.store	web, auth, mfa.verified
GET	`/admin/roles/{role}`	admin.roles.show	web, auth, mfa.verified
GET	`/admin/roles/{role}/edit`	admin.roles.edit	web, auth, mfa.verified
PATCH	`/admin/roles/{role}`	admin.roles.update	web, auth, mfa.verified
DELETE	`/admin/roles/{role}`	admin.roles.destroy	web, auth, mfa.verified
GET	`/admin/roles/{role}/permissions`	admin.roles.permissions	web, auth, mfa.verified
POST	`/admin/roles/{role}/permissions`	admin.roles.permissions.update	web, auth, mfa.verified
GET	`/admin/audit-logs`	admin.audit-logs.index	web, auth, mfa.verified
GET	`/admin/audit-logs/{auditLog}`	admin.audit-logs.show	web, auth, mfa.verified
GET	`/admin/audit-logs/export`	admin.audit-logs.export	web, auth, mfa.verified
GET	`/admin/settings`	admin.settings.index	web, auth, mfa.verified
GET	`/admin/settings/clinic`	admin.settings.clinic	web, auth, mfa.verified
POST	`/admin/settings/clinic`	admin.settings.clinic.update	web, auth, mfa.verified
GET	`/admin/settings/security`	admin.settings.security	web, auth, mfa.verified
POST	`/admin/settings/security`	admin.settings.security.update	web, auth, mfa.verified
GET	`/admin/backup`	admin.backup.index	web, auth, mfa.verified
POST	`/admin/backup`	admin.backup.create	web, auth, mfa.verified
POST	`/admin/backup/{backup}/restore`	admin.backup.restore	web, auth, mfa.verified
DELETE	`/admin/backup/{backup}`	admin.backup.destroy	web, auth, mfa.verified
GET	`/admin/sessions`	admin.sessions.index	web, auth, mfa.verified
DELETE	`/admin/sessions/{session}`	admin.sessions.terminate	web, auth, mfa.verified
GET	`/admin/security-dashboard`	admin.security-dashboard.index	web, auth, mfa.verified
GET	`/mfa/setup`	mfa.setup	web, auth

POST	`/mfa/setup`	mfa.setup.store	web, auth
GET	`/mfa/verify`	mfa.verify	web, auth
POST	`/mfa/verify`	mfa.verify.check	web, auth
GET	`/mfa/recovery`	mfa.recovery	web, auth
POST	`/mfa/recovery`	mfa.recovery.check	web, auth

5.8 FormRequest Validation

File: `app/Http/Requests/StoreUserRequest.php`

```php
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;
use Illuminate\Validation\Rules\Password;

class StoreUserRequest extends FormRequest
{
    public function authorize(): bool
    {
        return $this->user()->can('create', \App\Models\User::class);
    }

    public function rules(): array
    {
        $config = config('security.password');

        return [
            'nama' => 'required|string|max:255',
            'email' => 'required|email|unique:users,email|max:255',
            'no_telefon' => 'nullable|string|max:20',
            'jawatan' => 'nullable|string|max:100',
            'password' => [
                'required',
                'confirmed',
                Password::min($config['min_length'])
                    ->mixedCase()
                    ->numbers()
                    ->symbols(),
            ],
            'roles' => 'required|array|min:1',
            'roles.*' => 'exists:roles,id',
            'is_active' => 'boolean',
            'mfa_enforced' => 'boolean',
        ];
    }

    public function messages(): array
    {
        return [
            'nama.required' => 'Nama adalah wajib',
            'email.required' => 'Email adalah wajib',
            'email.unique' => 'Email ini sudah digunakan',
            'password.required' => 'Kata laluan adalah wajib',
            'password.confirmed' => 'Pengesahan kata laluan tidak sepadan',
            'roles.required' => 'Sekurang-kurangnya satu peranan diperlukan',
        ];
    }
}
```

---

6. Workflow dan User Flow

6.1 Login Workflow dengan MFA

```
[Pengguna] → Masukkan email & password
    ↓
[Sistem] Validate credentials
    ↓ (Berjaya)
[Sistem] Check if MFA enabled
    ↓ (Ya)
[Sistem] Check trusted device
    ↓ (Tidak trusted)
[Pengguna] → Masukkan TOTP code
    ↓
[Sistem] Verify TOTP
    ↓ (Berjaya)
[Sistem] Offer to trust device
    ↓
[Sistem] Create session
    ↓
[Pengguna] → Redirect ke dashboard
```

6.2 Password Change Workflow

```
[Pengguna] → Akses tukar password
    ↓
[Pengguna] → Masukkan current password
    ↓
[Sistem] Verify current password
    ↓ (Berjaya)
[Pengguna] → Masukkan new password (x2)
    ↓
[Sistem] Validate password rules
    ↓
[Sistem] Check password history
    ↓ (Tidak duplicate)
[Sistem] Update password
    ↓
[Sistem] Save to password history
    ↓
[Sistem] Log audit event
    ↓
[Sistem] Send email notification
    ↓
[Pengguna] → Force re-login dengan password baru
```

6.3 MFA Setup Workflow

```
[Pengguna] → Akses MFA setup
    ↓
[Sistem] Generate TOTP secret
    ↓
[Sistem] Display QR code
    ↓
[Pengguna] → Scan QR dengan authenticator app
    ↓
[Pengguna] → Masukkan verification code
    ↓
[Sistem] Verify code
    ↓ (Berjaya)
[Sistem] Generate 10 recovery codes
    ↓
[Sistem] Display recovery codes
    ↓
[Pengguna] → Simpan recovery codes
    ↓
[Sistem] Enable MFA
    ↓
[Sistem] Log audit event
```

6.4 Account Lockout Recovery

```
[Pengguna] → Gagal login 5 kali
    ↓
[Sistem] Lock account 30 minit
    ↓
[Sistem] Send alert email to Admin
    ↓
[Pengguna] → Request password reset
    ↓
[Sistem] Send reset link via email
    ↓
[Pengguna] → Click reset link
    ↓
[Sistem] Validate token
    ↓
[Pengguna] → Set new password
    ↓
[Sistem] Unlock account
    ↓
[Sistem] Reset failed attempts counter
```

6.5 Backup Workflow

```
[Admin] → Trigger manual backup
    ↓
[Sistem] Create backup log (status: pending)
    ↓
[Sistem] Start backup job (queue)
    ↓
[Sistem] Update status: running
```

```
    ↓
[Sistem] Dump database
    ↓
[Sistem] Compress files
    ↓
[Sistem] Encrypt backup
    ↓
[Sistem] Upload to local storage
    ↓
[Sistem] Upload to cloud storage
    ↓
[Sistem] Update status: completed
    ↓
[Sistem] Log audit event
    ↓
[Admin] → View backup in dashboard
```

6.6 State Management

User Status Flow:

```
[Pending] → [Active] → [Inactive] → [Active]
                ↓
           [Deleted (Soft)]
```

Account Lock Flow:

```
[Normal] → [Locked (Failed Login)] → [Normal (Timeout/Reset)]
```

MFA Status Flow:

```
[Disabled] → [Setup In Progress] → [Enabled] → [Disabled (Reset)]
```

Backup Status Flow:

```
[Pending] → [Running] → [Completed]
                ↓
            [Failed]
```

---

7. Keperluan UI/UX

7.1 Layout

- **Jenis Halaman**: Full page dengan sidebar navigation
- **Navigation**: Tambah ke menu Admin dengan sub-menu

Menu Structure:

```
📁 Tetapan & Keselamatan
├── 👥 Pengurusan Pengguna
├── 🔐 Peranan & Kebenaran
├── 📋 Log Audit
├── ⚙️ Tetapan Sistem
│   ├── Maklumat Klinik
│   ├── Keselamatan
│   └── Notifikasi
├── 💾 Backup & Recovery
├── 🖥️ Active Sessions
└── 📊 Security Dashboard
```

7.2 Bootstrap 5 + CoreUI Components

- ✅ **Card** - Dashboard widgets, settings sections
- ✅ **Table** - User list, audit logs, sessions, backup list
- ✅ **Form** - User form, settings form, role form
- ✅ **Modal** - Confirmation dialogs, MFA setup, force logout
- ✅ **Badge** - Status badges, role badges
- ✅ **Button** - Action buttons dengan icons
- ✅ **Tabs** - Settings categories
- ✅ **Toast** - Success/error notifications
- ✅ **Progress** - Backup progress
- ✅ **Alert** - Security warnings
- ✅ **Pagination** - Table pagination
- ✅ **Dropdown** - Bulk actions
- ✅ **Switch** - Toggle settings

7.3 Key UI Components

User List Page:

- Search bar dengan filter (role, status)
- DataTable dengan sorting dan pagination
- Action buttons (edit, view, toggle status, reset password)
- Bulk action dropdown (export, activate, deactivate)
- Add user button

User Form:

- Nama, email, telefon, jawatan
- Password fields dengan strength meter
- Role multi-select
- MFA enforcement toggle
- Active status switch

Role Permission Matrix:

- Grid view dengan modules as rows, permissions as columns
- Checkbox untuk setiap permission
- Select all per module / per permission
- Visual indicator untuk changes yang belum save

Audit Log View:

- Date range picker
- Filter by user, event type, category
- DataTable dengan color-coded event types
- Expandable rows untuk show old/new values
- Export buttons (PDF, CSV)

Security Dashboard:

- Real-time active users counter
- Login attempt chart (24h)
- Failed login alerts list
- System health gauges

- Recent audit events

MFA Setup:

- Step-by-step wizard
- QR code display
- Verification code input
- Recovery codes display dengan copy button
- Checkbox untuk confirm codes saved

7.4 Icons

Heroicons digunakan:

- `heroicon-o-users` - Pengguna
- `heroicon-o-shield-check` - Keselamatan
- `heroicon-o-key` - Peranan
- `heroicon-o-document-text` - Audit Log
- `heroicon-o-cog` - Tetapan
- `heroicon-o-cloud-arrow-up` - Backup
- `heroicon-o-computer-desktop` - Sessions
- `heroicon-o-chart-bar` - Dashboard
- `heroicon-o-lock-closed` - MFA
- `heroicon-o-exclamation-triangle` - Warnings
- `heroicon-o-check-circle` - Success
- `heroicon-o-x-circle` - Error/Failed

7.5 Responsive Design

- **Mobile Support**: Ya - simplified tables, stacked forms
- **Tablet Support**: Ya - adjusted grid layouts
- **Breakpoints**: Standard Bootstrap 5 (sm, md, lg, xl, xxl)

Mobile Considerations:

- Collapsible sidebar menu
- Card-based list view untuk tables
- Bottom sheet untuk actions
- Swipe gestures untuk navigation

---

8. Keperluan Keselamatan

8.1 Authentication & Authorization

- **Authentication**: Laravel Breeze dengan custom MFA layer
- **Middleware**: `auth`, `mfa.verified`, `ip.whitelist` untuk admin routes
- **Role-based Access**: Spatie Laravel-Permission

8.2 Data Protection (PDPA Compliance)

- **Audit Trail**: Semua operasi direkod dalam audit_logs
- **Soft Delete**: Semua pengguna soft-deleted, tidak permanent delete
- **Encryption**: Password di-hash dengan bcrypt, sensitive data dengan AES-256
- **Data Retention**: Audit logs disimpan 7 tahun

8.3 Input Validation & Security

- **CSRF Protection**: Semua forms dilindungi CSRF token
- **SQL Injection Prevention**: Guna Eloquent ORM, no raw queries
- **XSS Prevention**: Blade `{{ }}` escaping

- **Rate Limiting**: Laravel throttle middleware
- **Password Hashing**: bcrypt dengan cost factor 12

8.4 Additional Security Measures

- **Session Fixation Prevention**: Regenerate session on login
- **Clickjacking Protection**: X-Frame-Options header
- **Content Security Policy**: CSP headers
- **HTTPS Enforcement**: Force HTTPS in production
- **Secure Cookies**: HttpOnly, Secure flags
- **CORS**: Restricted to same origin

---

9. Keperluan Prestasi

9.1 Response Time

- **Dashboard Load**: < 2 saat
- **User List (100 users)**: < 1 saat
- **Audit Log Search**: < 3 saat
- **Login Process**: < 2 saat (termasuk MFA)
- **Backup Trigger**: < 5 saat untuk start, async completion

9.2 Scalability

- **Database Indexing**: Indexes pada semua search/filter fields
- **Query Optimization**: Eager loading untuk relationships
- **Caching**: Redis untuk sessions, settings, rate limiting
- **Pagination**: Default 15 records per page
- **Audit Log Partitioning**: Monthly partitions untuk performance

9.3 Concurrent Users

- **Expected Admin Users**: 5-10 concurrent
- **Session Storage**: Redis untuk scalability
- **Queue Processing**: Database queue dengan 3 workers

---

10. Keperluan Ujian

10.1 Unit Testing

File: `tests/Unit/Services/UserServiceTest.php`

- ✓ **Test**: Create user dengan valid data
- ✓ **Test**: Create user dengan duplicate email fails
- ✓ **Test**: Password validation rules enforced
- ✓ **Test**: Password history check prevents reuse
- ✓ **Test**: Role assignment works correctly
- ✓ **Test**: Deactivate user soft deletes
- ✓ **Test**: Cannot deactivate last Super Admin

File: `tests/Unit/Services/MfaServiceTest.php`

- ✓ **Test**: Generate TOTP secret
- ✓ **Test**: Verify valid TOTP code
- ✓ **Test**: Reject invalid TOTP code
- ✓ **Test**: Recovery code generation
- ✓ **Test**: Recovery code usage and invalidation

File: `tests/Unit/Services/AuditLogServiceTest.php`

- ✓ **Test**: Log creation dengan correct format
- ✓ **Test**: Sensitive fields are masked
- ✓ **Test**: Search and filter functionality
- ✓ **Test**: Export generates correct format

10.2 Feature Testing

File: `tests/Feature/UserManagementTest.php`

- ✓ **Test**: Super Admin can view user list
- ✓ **Test**: Super Admin can create user
- ✓ **Test**: Super Admin can update user
- ✓ **Test**: Super Admin can deactivate user
- ✓ **Test**: Admin cannot delete users
- ✓ **Test**: Admin cannot modify Super Admin
- ✓ **Test**: Unauthenticated user cannot access
- ✓ **Test**: User without MFA redirected to setup

File: `tests/Feature/AuthenticationTest.php`

- ✓ **Test**: Login dengan valid credentials
- ✓ **Test**: Login gagal dengan invalid credentials
- ✓ **Test**: Account locked after 5 failed attempts
- ✓ **Test**: Account unlocked after 30 minutes
- ✓ **Test**: MFA required for admin users
- ✓ **Test**: MFA bypass for trusted device
- ✓ **Test**: Password expiry forces change
- ✓ **Test**: Single session enforcement

File: `tests/Feature/AuditLogTest.php`

- ✓ **Test**: Login event logged
- ✓ **Test**: User CRUD events logged
- ✓ **Test**: Old/new values captured
- ✓ **Test**: Export functionality works

10.3 Integration Testing

- ✓ **Test**: Email notification delivery (password reset, alerts)
- ✓ **Test**: Backup job execution and cloud upload
- ✓ **Test**: TOTP integration dengan authenticator apps
- ✓ **Test**: Session management across multiple requests

10.4 User Acceptance Testing (UAT)

Scenario 1: Login dengan MFA

- Steps: Login → Enter TOTP → Trust device → Verify session created
- Expected Result: User logged in, audit log created, dashboard displayed

Scenario 2: Account Lockout & Recovery

- Steps: 5 failed logins → Account locked → Request reset → Set new password
- Expected Result: Account unlocked, can login dengan new password

Scenario 3: User Creation oleh Admin

- Steps: Create user → Assign roles → User receives email → User sets password
- Expected Result: User dapat login, permissions correct

Scenario 4: Audit Log Investigation

- Steps: Search by date range → Filter by user → View details → Export PDF
- Expected Result: Relevant logs displayed, export contains all data

Scenario 5: Manual Backup & Restore

- Steps: Trigger backup → Wait completion → Verify files → Test restore
- Expected Result: Backup successful, restore option available

---

11. Langkah Implementasi

11.1 Fasa 1: Database & Models (Minggu 1-2)

- [ ] Install Spatie Laravel-Permission package
- [ ] Install pragmarx/google2fa-laravel package
- [ ] Install spatie/laravel-backup package
- [ ] Create migration untuk extend users table
- [ ] Create migration untuk user_mfa table
- [ ] Create migration untuk trusted_devices table
- [ ] Create migration untuk password_histories table
- [ ] Create migration untuk login_attempts table
- [ ] Create migration untuk user_sessions table
- [ ] Create migration untuk audit_logs table (dengan partitioning)
- [ ] Create migration untuk ip_whitelists table
- [ ] Create migration untuk system_settings table
- [ ] Create migration untuk backup_logs table
- [ ] Create semua Model classes
- [ ] Create config files (security.php, audit.php, tetapan.php)
- [ ] Run migrations dan create seeders untuk default roles/permissions

11.2 Fasa 2: Repository & Service Layer (Minggu 3-4)

- [ ] Create UserRepository dengan CRUD methods
- [ ] Create UserService dengan business logic
- [ ] Create RoleRepository dan RoleService
- [ ] Create MfaService untuk TOTP handling
- [ ] Create AuditLogService untuk logging
- [ ] Create AuditLogRepository untuk queries
- [ ] Create SessionService untuk session management
- [ ] Create BackupService untuk backup operations
- [ ] Create SecurityService untuk security checks
- [ ] Create SettingsRepository dan SettingsService

11.3 Fasa 3: Middleware & Listeners (Minggu 5)

- [ ] Create EnforceMfa middleware
- [ ] Create CheckIpWhitelist middleware
- [ ] Create SingleSession middleware

- [ ] Create AuditActivity middleware
- [ ] Create LogSuccessfulLogin listener
- [ ] Create LogFailedLogin listener
- [ ] Create LogLogout listener
- [ ] Create SendSecurityAlert listener
- [ ] Register middleware dalam Kernel
- [ ] Register listeners dalam EventServiceProvider

11.4 Fasa 4: FormRequest Validation (Minggu 5)

- [ ] Create StoreUserRequest
- [ ] Create UpdateUserRequest
- [ ] Create StoreRoleRequest
- [ ] Create UpdateRoleRequest
- [ ] Create VerifyMfaRequest
- [ ] Create UpdateSettingsRequest
- [ ] Create custom password validation rule

11.5 Fasa 5: Controllers & Routes (Minggu 6-7)

- [ ] Create UserController dengan Route Attributes
- [ ] Create RoleController dengan Route Attributes
- [ ] Create PermissionController
- [ ] Create AuditLogController
- [ ] Create SettingsController
- [ ] Create BackupController
- [ ] Create SessionController
- [ ] Create SecurityDashboardController
- [ ] Create MfaController
- [ ] Clear route cache dan verify routes

11.6 Fasa 6: Views & UI (Minggu 8-10)

- [ ] Create user management views (index, create, edit, show)
- [ ] Create role management views (index, create, edit, permissions)
- [ ] Create audit log views (index, show)
- [ ] Create settings views (clinic, security, notifications)
- [ ] Create backup views (index, restore)
- [ ] Create session management view
- [ ] Create security dashboard view
- [ ] Create MFA views (setup, verify, recovery)
- [ ] Implement responsive design
- [ ] Add JavaScript untuk interactivity

11.7 Fasa 7: Notifications & Jobs (Minggu 11)

- [ ] Create MfaSetupNotification
- [ ] Create PasswordResetNotification
- [ ] Create AccountLockedNotification
- [ ] Create NewDeviceLoginNotification
- [ ] Create BackupFailedNotification
- [ ] Create RunScheduledBackup command

- [ ] Create PurgeOldAuditLogs command
- [ ] Create CleanExpiredSessions command
- [ ] Configure queue workers
- [ ] Configure task scheduler

11.8 Fasa 8: Testing (Minggu 12-13)

- [ ] Write unit tests untuk services
- [ ] Write feature tests untuk all endpoints
- [ ] Write integration tests
- [ ] Perform manual UAT
- [ ] Security testing (penetration testing basic)
- [ ] Performance testing
- [ ] Fix bugs dan issues

11.9 Fasa 9: Deployment & Training (Minggu 14)

- [ ] Deploy ke staging environment
- [ ] UAT dengan actual users
- [ ] Fix final issues
- [ ] Deploy ke production
- [ ] Create user documentation
- [ ] Training session untuk Admin
- [ ] Monitor logs untuk issues

---

12. Kriteria Kejayaan

12.1 Metrics Utama

- **Login Success Rate**: > 99% untuk legitimate users
- **MFA Adoption**: 100% untuk Admin roles
- **Audit Log Coverage**: 100% untuk semua operasi kritikal
- **Backup Success Rate**: > 99%
- **Zero Security Incidents**: Tiada unauthorized access

12.2 User Satisfaction

- **Admin Satisfaction**: > 4.0/5.0 (survey)
- **Ease of Use**: Admin boleh manage users tanpa rujuk manual
- **Response Time**: Tiada complaint tentang slowness

12.3 Technical Metrics

- **Uptime**: > 99.5%
- **Response Time**: < 2 saat untuk 95% requests
- **Bug Rate**: < 3 bugs per bulan selepas deployment
- **Test Coverage**: > 80%

---

13. Risks & Mitigation

Risk	Likelihood	Impact	Mitigation
MFA adoption resistance	Medium	Medium	Provide clear documentation, training, dan trusted device feature

Audit log storage growth	High	Low	Implement partitioning, archival strategy, dan monitoring
Backup failure undetected	Low	High	Automated monitoring, alerts, dan regular restore testing
Password policy complaints	Medium	Low	Clear communication, password manager recommendation
Session timeout frustration	Medium	Low	Configurable timeout, warning before logout
IP whitelist too restrictive	Low	Medium	Easy self-service untuk add IP, bypass untuk emergency
Performance degradation from audit logging	Medium	Medium	Async logging via queue, optimized queries
Recovery code loss	Medium	High	Clear instructions, Admin reset option

---

14. Dependencies

14.1 External Packages

- ☑ **spatie/laravel-permission**: ^6.0 - RBAC management
- ☑ **pragmarx/google2fa-laravel**: ^2.0 - TOTP generation dan verification
- ☑ **spatie/laravel-backup**: ^8.0 - Database backup
- ☑ **bacon/bacon-qr-code**: ^2.0 - QR code generation untuk MFA setup
- ☑ **jenssegers/agent**: ^2.6 - Device detection untuk sessions
- ☑ **maatwebsite/excel**: ^3.1 - User bulk import

14.2 Related Features/Modules

Bergantung Kepada:

- Laravel Authentication (Breeze) - mesti setup dahulu
- Email configuration - untuk notifications
- Queue configuration - untuk async jobs
- Redis configuration - untuk sessions dan rate limiting

Memberi Impak Kepada:

- Semua modul lain - semua akan guna permission checks
- Semua modul lain - semua operasi akan di-audit

14.3 Third-Party Integrations

- ☑ **SMTP Service**: Untuk email notifications
  - Configuration: MAIL_* environment variables
- ☑ **Cloud Storage (Optional)**: S3/GCS untuk offsite backup
  - Configuration: AWS_* atau GCS_* environment variables

---

15. Acceptance Criteria

15.1 Functional Acceptance

- ☑ Super Admin boleh CRUD pengguna dengan semua fields
- ☑ Super Admin boleh assign multiple roles kepada pengguna
- ☑ Roles boleh dibuat dengan permissions granular per module

- ✓ MFA wajib untuk Super Admin dan Admin roles
- ✓ TOTP verification berfungsi dengan Google Authenticator
- ✓ Recovery codes boleh digunakan jika authenticator tidak available
- ✓ Trusted device feature berfungsi (skip MFA 30 hari)
- ✓ Semua login/logout direkod dalam audit log
- ✓ Semua CRUD operations direkod dengan old/new values
- ✓ Audit log boleh dicari dan di-filter
- ✓ Audit log boleh di-export ke PDF/CSV
- ✓ Password policy dikuatkuasakan (12 chars, complexity)
- ✓ Password history mencegah reuse (5 passwords)
- ✓ Account locked selepas 5 failed attempts
- ✓ IP whitelist menyekat akses admin dari IP tidak sah
- ✓ Single session policy berfungsi
- ✓ Session timeout auto-logout berfungsi
- ✓ Manual dan scheduled backup berfungsi
- ✓ Backup boleh di-restore
- ✓ Security dashboard menunjukkan real-time metrics

15.2 Technical Acceptance

- ✓ Semua feature tests lulus
- ✓ Semua unit tests lulus
- ✓ Kod mengikut conventions dari DEVELOPER_GUIDE.md
- ✓ Kod diformat dengan ./vendor/bin/pint
- ✓ Tiada N+1 query problems
- ✓ Route cache cleared selepas tambah routes
- ✓ Audit log menggunakan partitioning
- ✓ Sensitive data encrypted dalam database

15.3 Quality Acceptance

- ✓ Kod di-review oleh peer
- ✓ Manual testing selesai untuk all scenarios
- ✓ Tiada console errors atau warnings
- ✓ Responsive design berfungsi di mobile/tablet
- ✓ Error messages jelas dan dalam Bahasa Malaysia

15.4 Documentation Acceptance

- ✓ PRD dikemaskini dengan implementation notes
- ✓ User guide untuk Admin tersedia
- ✓ API documentation untuk MFA endpoints
- ✓ Security configuration guide

---

16. Lampiran

16.1 Default Roles & Permissions Matrix

Module	Permission	Super Admin	Admin	Doktor	Jururawat	Kerani	Farmasi
Pendaftaran	view	✓	✓	✓	✓	✓	✓

Pendaftaran	create	✓	✓	-	-	✓	-
Pendaftaran	update	✓	✓	-	-	✓	-
Pendaftaran	delete	✓	-	-	-	-	-
EMR	view	✓	-	✓	✓	-	-
EMR	create	✓	-	✓	-	-	-
EMR	update	✓	-	✓	-	-	-
Farmasi	view	✓	✓	✓	✓	-	✓
Farmasi	dispense	✓	-	-	-	-	✓
Billing	view	✓	✓	-	-	✓	-
Billing	create	✓	✓	-	-	✓	-
Billing	refund	✓	✓	-	-	-	-
Tetapan	view	✓	✓	-	-	-	-
Tetapan	update	✓	-	-	-	-	-
Pengguna	view	✓	✓	-	-	-	-
Pengguna	create	✓	✓	-	-	-	-
Pengguna	delete	✓	-	-	-	-	-
Audit	view	✓	✓	-	-	-	-
Backup	create	✓	-	-	-	-	-
Backup	restore	✓	-	-	-	-	-

16.2 Audit Event Types

Category	Event Type	Description
Authentication	`login`	User berjaya login
Authentication	`logout`	User logout
Authentication	`login_failed`	Percubaan login gagal
Authentication	`mfa_setup`	User setup MFA
Authentication	`mfa_verified`	MFA verification berjaya
Authentication	`mfa_failed`	MFA verification gagal
Authentication	`password_changed`	User tukar password
Authentication	`password_reset`	Password reset dilakukan
User Management	`user_created`	Pengguna baharu dicipta
User Management	`user_updated`	Maklumat pengguna dikemaskini
User Management	`user_deactivated`	Pengguna dinyahaktifkan

User Management	`user_activated`	Pengguna diaktifkan semula
User Management	`user_deleted`	Pengguna dipadam
Role Management	`role_created`	Role baharu dicipta
Role Management	`role_updated`	Role dikemaskini
Role Management	`role_deleted`	Role dipadam
Role Management	`permission_assigned`	Permission assigned ke role
Role Management	`permission_revoked`	Permission revoked dari role
Session	`session_started`	Session baharu bermula
Session	`session_ended`	Session tamat (logout)
Session	`session_terminated`	Session ditamatkan paksa
Session	`session_timeout`	Session tamat kerana timeout
Security	`account_locked`	Akaun dikunci
Security	`account_unlocked`	Akaun dibuka semula
Security	`ip_blocked`	IP disekat
Security	`suspicious_activity`	Aktiviti mencurigakan dikesan
System	`settings_updated`	Tetapan sistem dikemaskini
System	`backup_created`	Backup dicipta
System	`backup_restored`	Backup dipulihkan
System	`backup_failed`	Backup gagal
Data Access	`data_exported`	Data diexport
Data Access	`report_generated`	Laporan dijana
Data Access	`bulk_import`	Import data secara bulk

16.3 System Settings Keys

Group	Key	Type	Description
clinic	nama	string	Nama klinik
clinic	alamat	string	Alamat penuh
clinic	no_telefon	string	Nombor telefon
clinic	email	string	Email klinik
clinic	logo	string	Path to logo file
clinic	waktu_operasi	json	Waktu operasi mengikut hari
security	session_timeout	integer	Idle timeout dalam minit
security	password_expiry_days	integer	Password expiry period

security	max_login_attempts	integer	Max failed attempts
security	lockout_minutes	integer	Lockout duration
security	mfa_enforced_roles	json	Roles yang wajib MFA
backup	schedule	string	daily/weekly
backup	time	string	Backup time (HH:MM)
backup	retention_local	integer	Local retention days
backup	retention_cloud	integer	Cloud retention days
backup	notify_emails	json	Email untuk notification

16.4 Change Log

Tarikh	Penulis	Perubahan
13 Januari 2026	AI Assistant	PRD awal dicipta

16.5 Approval

- [ ] **Product Owner**: ________________ - ________________
- [ ] **Tech Lead**: ________________ - ________________
- [ ] **Pengurus Klinik**: ________________ - ________________
- [ ] **Stakeholders**: ________________ - ________________

Status Implementasi: Belum Bermula **Tarikh Selesai**: TBD

Catatan: Dokumen ini adalah living document dan akan dikemaskini mengikut keperluan semasa development.