

PRD: Sistem Pengurusan Maklumat Pesakit

PRD ID: KLINIK-PendaftaranPesakit-PR2026-01-pengurusan-maklumat-pesakit **Dicipta:** 12 Januari 2026 **Pengarang:** Pasukan Pembangunan **Dikemaskini:** 12 Januari 2026 **Status:** Perancangan

Format Penamaan PRD

- KLINIK-PendaftaranPesakit** - Modul sistem klinik swasta untuk pendaftaran pesakit
- PR** - Awalan tetap untuk "Product Requirement"
- 2026** - Tahun empat digit
- 01** - Nombor jujukan dua digit bermula dari 01 untuk setiap kombinasi modul-submodul setahun
- pengurusan-maklumat-pesakit** - Nama ciri dalam format kebab-case

1. Gambaran Keseluruhan

1.1 Ringkasan Ciri

Sistem Pengurusan Maklumat Pesakit adalah modul teras untuk pendaftaran dan pengurusan data pesakit di Klinik Al-Huda. Sistem ini membolehkan kakitangan kaunter (Kerani dan Jururawat) mendaftar pesakit baharu, mengemaskini maklumat sedia ada, dan mencari rekod pesakit dengan cepat dan tepat. Sistem dilengkapi dengan pengesahan data automatik, pencegahan rekod pendua, dan pematuhan penuh kepada PDPA.

1.2 Metadata

- Nama Ciri:** Sistem Pengurusan Maklumat Pesakit
- Modul/SubModul:** Admin/Pesakit/FrontDesk/PendaftaranPesakit
- Peranan Sasaran:** Kerani, Jururawat
- Keutamaan:** Tinggi
- Status:** Perancangan
- Anggaran Usaha:** Sederhana (2-3 minggu)

2. Pernyataan Masalah

2.1 Situasi Semasa

Klinik Al-Huda menghadapi cabaran dalam pengurusan maklumat pesakit:

- Rekod pendua:** Pesakit yang sama didaftarkan berkali-kali dengan IC berbeza atau typo dalam nama
- Proses pendaftaran lambat:** Kerani perlu memasukkan data secara manual tanpa bantuan auto-validation
- Tiada standard data:** Format IC, nombor telefon, dan alamat tidak konsisten
- Sukar cari pesakit:** Carian tidak efisien, perlu scroll senarai panjang
- Tiada audit trail:** Tidak dapat kesan siapa ubah data pesakit dan bila
- Risiko PDPA:** Tiada mekanisme consent management yang sistematik

2.2 Hasil Yang Diingini

Selepas sistem ini dilaksanakan:

- ✔ Pendaftaran pesakit baharu dalam masa kurang dari 3 minit
- ✔ Tiada rekod pendua dengan IC yang sama
- ✔ Auto-validation format IC, telefon, dan email mengurangkan kesilapan data
- ✔ Carian pesakit instant dengan berbilang kriteria (IC, nama, telefon, ID)
- ✔ Audit trail penuh untuk setiap perubahan data pesakit
- ✔ Pematuhan PDPA dengan consent management automatik
- ✔ Data pesakit standard dan berkualiti tinggi untuk integrasi masa hadapan

2.3 Metrik Kejayaan

Metrik	Sasaran	Cara Ukur
Masa pendaftaran pesakit baharu	< 3 minit	Masa dari mula isi form hingga ID dijana
Kadar rekod pendua	< 0.5%	Peratus pesakit dengan IC sama berbanding jumlah pesakit
Ketepatan data	> 98%	Audit manual 100 rekod setiap bulan
Masa carian pesakit	< 5 saat	Masa dari mula taip hingga hasil carian dipapar
Pematuhan PDPA	100%	Semua pesakit baharu ada rekod consent
Pengguna berpuas hati	> 90%	Survey kepuasan Kerani dan Jururawat setiap 3 bulan

3. User Stories

3.1 User Stories Utama

- **Sebagai** Kerani kaunter, **saya mahu** mendaftar pesakit baharu dengan cepat dan mudah **supaya** pesakit tidak perlu menunggu lama dan data rekod adalah tepat
- **Sebagai** Jururawat, **saya mahu** mengesahkan data pesakit yang didaftarkan oleh Kerani **supaya** memastikan tiada kesilapan data sebelum pesakit berjumpa doktor
- **Sebagai** Kerani/Jururawat, **saya mahu** mencari pesakit sedia ada dengan cepat menggunakan IC, nama, atau telefon **supaya** dapat akses rekod pesakit tanpa perlu scroll senarai panjang
- **Sebagai** Kerani/Jururawat, **saya mahu** mengemaskini maklumat pesakit (contoh: alamat, telefon) **supaya** rekod sentiasa terkini untuk komunikasi dan billing
- **Sebagai** Kerani, **bila** saya mendaftar pesakit dengan IC yang sudah wujud, **saya sepatutnya** menerima amaran dan pilihan untuk guna rekod sedia ada atau buat baharu
- **Sebagai** Pemilik Klinik, **saya mahu** setiap pendaftaran pesakit mempunyai rekod consent untuk kegunaan data **supaya** klinik mematuhi Akta Perlindungan Data Peribadi 2010

3.2 Edge Cases & User Stories Sekunder

- **Sebagai** Kerani, **bila** pesakit adalah warganegara asing tanpa IC Malaysia, **saya sepatutnya** boleh mendaftar menggunakan nombor Passport
- **Sebagai** Kerani, **bila** pesakit adalah kanak-kanak bawah 12 tahun tanpa IC, **saya sepatutnya** boleh mendaftar menggunakan Sijil Kelahiran dan maklumat penjaga
- **Sebagai** Jururawat, **bila** pesakit sudah approved tetapi ada kesilapan data, **saya sepatutnya** boleh edit dengan log audit untuk perubahan
- **Sebagai** Admin, **saya mahu** tandakan pesakit sebagai "Tidak Aktif" (contoh: pindah, meninggal) **supaya** rekod tidak dipadam (PDPA retention) tetapi tidak muncul dalam carian biasa

4. Keperluan Fungsi

4.1 Ciri-ciri Teras

- **FR-01:** Sistem mesti membenarkan pendaftaran pesakit baharu dengan maklumat asas (Nama, IC/Passport, DOB, Jantina, Telefon, Alamat)

- **FR-02:** Sistem mesti auto-validate format IC Malaysia (12 digit YYMMDD-PB-XXXX) dan auto-detect tarikh lahir + jantina dari IC
- **FR-03:** Sistem mesti menjana ID Pesakit unik secara automatik dalam format P{YYYY}-{NNNN} (contoh: P2026-0001)
- **FR-04:** Sistem mesti menyemak rekod pendua berdasarkan IC/Passport sebelum simpan
- **FR-05:** Sistem mesti papar amaran jika IC/Passport sudah wujud dan beri pilihan: guna rekod lama atau buat baharu
- **FR-06:** Sistem mesti rekod consent PDPA untuk setiap pesakit baharu dengan timestamp
- **FR-07:** Sistem mesti membenarkan carian pesakit menggunakan IC, Nama, No. Telefon, atau ID Pesakit
- **FR-08:** Sistem mesti membenarkan Kerani mendaftar pesakit dalam status "Draft"
- **FR-09:** Sistem mesti membenarkan Jururawat verify dan approve pesakit dari "Draft" ke "Aktif"
- **FR-10:** Sistem mesti log semua perubahan data pesakit (audit trail) dengan user ID dan timestamp
- **FR-11:** Sistem mesti membenarkan kemaskini maklumat pesakit (alamat, telefon, email, dll) selepas pendaftaran
- **FR-12:** Sistem mesti sokong soft delete (data tidak dipadam sepenuhnya untuk pematuhan retention)
- **FR-13:** Sistem mesti papar senarai pesakit dengan pagination (15 rekod per page)
- **FR-14:** Sistem mesti membenarkan export senarai pesakit ke Excel/CSV (untuk Jururawat sahaja)

4.2 Kebenaran & Kawalan Akses

Peranan Yang Diperlukan:

- Kerani (Front Desk Staff)
- Jururawat (Nurse)

Kebenaran Khusus:

Fungsi	Kerani	Jururawat
Daftar pesakit baharu (Draft)	✓	✓
Verify & Approve pesakit	×	✓
Lihat senarai pesakit	✓	✓
Cari pesakit	✓	✓
Kemaskini maklumat pesakit	✓	✓
Tandakan tidak aktif	×	✓
Export data pesakit	×	✓
Lihat audit trail	×	✓

Logik Policy/Gate:

- Kerani hanya boleh create draft dan edit pesakit yang belum di-approve
- Jururawat boleh approve, edit mana-mana pesakit, dan lihat audit trail
- Semua perubahan data pesakit mesti log dalam audit trail

4.3 Sistem Badge (Tidak Berkenaan)

Ciri ini tidak memerlukan badge notification kerana ia adalah modul CRUD standard tanpa approval workflow kompleks yang memerlukan notifikasi real-time.

4.4 Pengesahan Data

Medan Wajib:

- Nama Penuh
- No. IC/Passport
- Tarikh Lahir (auto dari IC atau manual)
- Jantina (auto dari IC atau manual)

- No. Telefon
- Alamat Surat Menyurat
- Poskod
- Bandar
- Negeri
- Waris Terdekat - Nama
- Waris Terdekat - No. Telefon

Peraturan Pengesahan:

Medan	Peraturan
Nama Penuh	Required, string, max 255 characters
No. IC	Required jika Warganegara Malaysia, format 12 digit tanpa tanda, unique
No. Passport	Required jika warganegara asing, string, max 20 characters, unique
Tarikh Lahir	Required, date, format YYYY-MM-DD, must be past date
Jantina	Required, enum (Lelaki, Perempuan)
No. Telefon	Required, format 01X-XXXXXXX atau 01X-XXXXXXX (Malaysian mobile), unique
Email	Optional, email format, unique if provided
Alamat	Required, string, max 500 characters
Poskod	Required, 5 digits
Bandar	Required, string, max 100 characters
Negeri	Required, dropdown (list 14 negeri + WP)
Kumpulan Darah	Optional, enum (A+, A-, B+, B-, AB+, AB-, O+, O-)
Alahan	Optional, text
Waris Nama	Required, string, max 255 characters
Waris Telefon	Required, format sama seperti No. Telefon

Peraturan Perniagaan:

- **BR-01:** IC/Passport mesti unique dalam sistem (trigger duplicate detection)
- **BR-02:** Jika IC format Malaysia, auto-extract DOB dan Jantina, user boleh override jika ada kesilapan
- **BR-03:** Pesakit bawah 18 tahun mesti ada maklumat Waris Terdekat
- **BR-04:** No. Telefon mesti format Malaysia (01X-XXXXXXX), sistem auto-format (remove spaces, dashes)
- **BR-05:** Email jika diisi, mesti unique (satu email satu pesakit sahaja)
- **BR-06:** Status default pesakit baharu: "Draft" jika daftar oleh Kerani, "Aktif" jika daftar oleh Jururawat
- **BR-07:** Pesakit tidak boleh deleted permanently (soft delete sahaja untuk PDPA compliance)
- **BR-08:** Consent PDPA mesti rekod untuk setiap pesakit baharu (timestamp + user yang daftar)

5. Spesifikasi Teknikal

5.1 Arkitektur

Struktur Modul

```
Admin/
└─ Pesakit/
  └─ FrontDesk/
    └─ PendaftaranPesakit/
      ├── PesakitController.php (Laravel Controller)
      ├── index.blade.php (Senarai pesakit)
      ├── create.blade.php (Form pendaftaran)
      ├── edit.blade.php (Form kemaskini)
      ├── show.blade.php (Lihat detail pesakit)
      └── search.blade.php (Carian pesakit)
```

Path Modul: app/Http/Controllers/Admin/PesakitController.php **Jenis Komponen:** Standard Laravel Controller + Blade Templates **Nama Komponen:** PesakitController (sudah wujud sebagai stub)

Arahan Untuk Generate (Jika Perlu)

```
# Controller sudah wujud, perlu implement sahaja
# FormRequests
php artisan make:request StorePesakitRequest
php artisan make:request UpdatePesakitRequest

# Service & Repository
php artisan make:class Services/PesakitService
php artisan make:class Repositories/PesakitRepository

# Model migration
php artisan make:model Pesakit -m
```

5.2 Skema Pangkalan Data

Jadual Baharu

Jadual: pesakits

Column	Type	Attributes	Description
id	BIGINT UNSIGNED	PRIMARY KEY, AUTO_INCREMENT	ID unik pesakit (internal)
patient_id	VARCHAR(20)	UNIQUE, NOT NULL	ID pesakit (user-facing, format: P2026-0001)
ic_no	VARCHAR(12)	UNIQUE, NULLABLE	No. IC (12 digit tanpa dash)
passport_no	VARCHAR(20)	UNIQUE, NULLABLE	No. Passport (untuk warganegara asing)
name	VARCHAR(255)	NOT NULL	Nama penuh pesakit
date_of_birth	DATE	NOT NULL	Tarikh lahir
gender	ENUM('Lelaki', 'Perempuan')	NOT NULL	Jantina
phone_primary	VARCHAR(15)	NOT NULL	No. telefon utama
phone_alternative	VARCHAR(15)	NULLABLE	No. telefon alternatif

email	VARCHAR(255)	UNIQUE, NULLABLE	Email
address	TEXT	NOT NULL	Alamat surat menyurat
postcode	VARCHAR(5)	NOT NULL	Poskod
city	VARCHAR(100)	NOT NULL	Bandar
state	VARCHAR(50)	NOT NULL	Negeri
country	VARCHAR(100)	DEFAULT 'Malaysia'	Negara
blood_type	ENUM('A+', 'A-', 'B+', 'B-', 'AB+', 'AB-', 'O+', 'O-')	NULLABLE	Kumpulan darah
allergies	TEXT	NULLABLE	Alahan
emergency_contact_name	VARCHAR(255)	NOT NULL	Nama waris terdekat
emergency_contact_phone	VARCHAR(15)	NOT NULL	No. telefon waris
emergency_contact_relationship	VARCHAR(50)	NULLABLE	Hubungan dengan waris
status	ENUM('Draft', 'Aktif', 'Tidak Aktif')	DEFAULT 'Draft'	Status pesakit
pdpa_consent	BOOLEAN	DEFAULT FALSE	Consent PDPA (TRUE jika setuju)
pdpa_consent_at	TIMESTAMP	NULLABLE	Timestamp consent PDPA
pdpa_consent_by	BIGINT UNSIGNED	NULLABLE	User ID yang rekod consent
registered_by	BIGINT UNSIGNED	NULLABLE	User ID yang daftar pesakit
approved_by	BIGINT UNSIGNED	NULLABLE	User ID yang approve pesakit
approved_at	TIMESTAMP	NULLABLE	Timestamp approval
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Tarikh dicipta
updated_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP ON UPDATE	Tarikh dikemaskini
deleted_at	TIMESTAMP	NULLABLE	Soft delete timestamp

Indexes:

```

PRIMARY KEY (id)
UNIQUE KEY unique_patient_id (patient_id)
UNIQUE KEY unique_ic_no (ic_no) WHERE ic_no IS NOT NULL
UNIQUE KEY unique_passport_no (passport_no) WHERE passport_no IS NOT NULL
UNIQUE KEY unique_email (email) WHERE email IS NOT NULL
INDEX idx_name (name)
INDEX idx_phone (phone_primary)

```

```
INDEX idx_status (status)
INDEX idx_created_at (created_at)
```

Foreign Keys:

```
FOREIGN KEY (registered_by) REFERENCES users(id) ON DELETE SET NULL
FOREIGN KEY (approved_by) REFERENCES users(id) ON DELETE SET NULL
FOREIGN KEY (pdpa_consent_by) REFERENCES users(id) ON DELETE SET NULL
```

Jadual: `pesakit_audit_logs` (Audit Trail)

Column	Type	Attributes	Description
id	BIGINT UNSIGNED	PRIMARY KEY, AUTO_INCREMENT	ID log
pesakit_id	BIGINT UNSIGNED	NOT NULL	ID pesakit yang diubah
user_id	BIGINT UNSIGNED	NOT NULL	User yang buat perubahan
action	ENUM('create', 'update', 'approve', 'deactivate')	NOT NULL	Jenis tindakan
old_values	JSON	NULLABLE	Nilai lama (untuk update)
new_values	JSON	NULLABLE	Nilai baharu (untuk update)
ip_address	VARCHAR(45)	NULLABLE	IP address user
user_agent	TEXT	NULLABLE	Browser user agent
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Timestamp log

Indexes:

```
PRIMARY KEY (id)
INDEX idx_pesakit_id (pesakit_id)
INDEX idx_user_id (user_id)
INDEX idx_created_at (created_at)
FOREIGN KEY (pesakit_id) REFERENCES pesakits(id) ON DELETE CASCADE
FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE SET NULL
```

Pengubahsuaian Jadual Sedia Ada

Tiada. Sistem ini adalah modul baharu tanpa perubahan pada jadual sedia ada.

5.3 Model Eloquent

Model Baharu: `Pesakit`

Path: `app/Models/Pesakit.php`

Relationships:

- `belongsTo(User::class, 'registered_by')` - User yang daftar
- `belongsTo(User::class, 'approved_by')` - User yang approve
- `belongsTo(User::class, 'pdpa_consent_by')` - User yang rekod consent

- `hasMany(PesakitAuditLog::class)` - Audit trail logs

Casts:

```
'date_of_birth' => 'date',
'pdpa_consent' => 'boolean',
'pdpa_consent_at' => 'datetime',
'approved_at' => 'datetime',
'created_at' => 'datetime',
'updated_at' => 'datetime',
'deleted_at' => 'datetime',
'old_values' => 'array',
'new_values' => 'array',
```

Fillable:

```
[
    'patient_id', 'ic_no', 'passport_no', 'name', 'date_of_birth', 'gender',
    'phone_primary', 'phone_alternative', 'email',
    'address', 'postcode', 'city', 'state', 'country',
    'blood_type', 'allergies',
    'emergency_contact_name', 'emergency_contact_phone', 'emergency_contact_relationship',
    'status', 'pdpa_consent', 'pdpa_consent_at', 'pdpa_consent_by',
    'registered_by', 'approved_by', 'approved_at'
]
```

Scopes:

```
public function scopeAktif($query) {
    return $query->where('status', 'Aktif');
}

public function scopeDraft($query) {
    return $query->where('status', 'Draft');
}

public function scopeTidakAktif($query) {
    return $query->where('status', 'Tidak Aktif');
}
```

Accessors:

```
public function getFormattedIcAttribute() {
    if ($this->ic_no) {
        return substr($this->ic_no, 0, 6) . '-' . substr($this->ic_no, 6, 2) . '-' . substr($this->ic_no, 8);
    }
    return null;
}

public function getAgeAttribute() {
    return $this->date_of_birth->age;
}

public function getStatusBadgeAttribute() {
```

```

return match($this->status) {
    'Aktif' => '<span class="badge badge-success">Aktif</span>',
    'Draft' => '<span class="badge badge-warning">Draft</span>',
    'Tidak Aktif' => '<span class="badge badge-secondary">Tidak Aktif</span>',
};
}

```

Boot Method (Auto-generate Patient ID):

```

protected static function boot() {
    parent::boot();

    static::creating(function ($pesakit) {
        if (empty($pesakit->patient_id)) {
            $year = date('Y');
            $lastId = static::whereYear('created_at', $year)->max('id') ?? 0;
            $pesakit->patient_id = 'P' . $year . '-' . str_pad($lastId + 1, 4, '0', STR_PAD_LEFT);
        }

        // Auto-extract DOB and Gender from IC if IC provided
        if ($pesakit->ic_no && strlen($pesakit->ic_no) == 12) {
            // Extract DOB
            $year = substr($pesakit->ic_no, 0, 2);
            $month = substr($pesakit->ic_no, 2, 2);
            $day = substr($pesakit->ic_no, 4, 2);
            $fullYear = ($year > 50) ? "19$year" : "20$year";
            $pesakit->date_of_birth = "$fullYear-$month-$day";

            // Extract Gender (last digit: odd = male, even = female)
            $lastDigit = (int)substr($pesakit->ic_no, -1);
            $pesakit->gender = ($lastDigit % 2 == 0) ? 'Perempuan' : 'Lelaki';
        }
    });
}

```

Factory: Ya (untuk testing) **Seeder:** Ya (sample data untuk development)

Model Baru: `PesakitAuditLog`

Path: `app/Models/PesakitAuditLog.php`

Relationships:

- `belongsTo(Pesakit::class, 'pesakit_id')`
- `belongsTo(User::class, 'user_id')`

Fillable:

```
['pesakit_id', 'user_id', 'action', 'old_values', 'new_values', 'ip_address', 'user_agent']
```

5.4 Routes

Routes sudah partially wujud dalam: `app/Http/Controllers/Admin/PesakitController.php`

Routes yang perlu ditambah:

```
// Dalam PesakitController dengan Route Attributes

#[Get('/', name: 'admin.pesakit.index')]
public function index(Request $request)

#[Get('/create', name: 'admin.pesakit.create')]
public function create()

#[Post('/', name: 'admin.pesakit.store')]
public function store(StorePesakitRequest $request)

#[Get('/{pesakit}', name: 'admin.pesakit.show')]
public function show(Pesakit $pesakit)

#[Get('/{pesakit}/edit', name: 'admin.pesakit.edit')]
public function edit(Pesakit $pesakit)

#[Patch('/{pesakit}', name: 'admin.pesakit.update')]
public function update(UpdatePesakitRequest $request, Pesakit $pesakit)

#[Delete('/{pesakit}', name: 'admin.pesakit.destroy')]
public function destroy(Pesakit $pesakit)

#[Get('/search', name: 'admin.pesakit.search')]
public function search(Request $request)

#[Patch('/{pesakit}/approve', name: 'admin.pesakit.approve')]
public function approve(Pesakit $pesakit)

#[Patch('/{pesakit}/deactivate', name: 'admin.pesakit.deactivate')]
public function deactivate(Pesakit $pesakit)

#[Get('/export', name: 'admin.pesakit.export')]
public function export()

#[Get('/{pesakit}/audit-log', name: 'admin.pesakit.audit')]
public function auditLog(Pesakit $pesakit)
```

Route Middleware: ['web', 'auth'] (sudah ada dalam PesakitController)

5.5 Komponen UI

Layout

- **Jenis Halaman:** Full page (menggunakan admin layout sedia ada)
- **Navigasi:** Tambah dalam sidebar admin
 - Menu: "Pesakit"
 - Icon: `heroicon-o-users`
 - Submenu:
 - "Senarai Pesakit" → `admin.pesakit.index`
 - "Daftar Baharu" → `admin.pesakit.create`
 - "Carian" → `admin.pesakit.search`

Bootstrap 5 + CoreUI Components

Komponen UI yang digunakan (mengikut stack sedia ada):

Form Components:

- `<input class="form-control">` - Input text standard
- `<select class="form-select">` - Dropdown
- `<textarea class="form-control">` - Text area untuk alamat
- `<div class="form-check">` - Checkbox untuk PDPA consent
- `<button class="btn btn-primary">` - Submit button
- `<button class="btn btn-secondary">` - Cancel button

Table Components:

- `<table class="table table-striped table-hover">` - Senarai pesakit
- Pagination: Laravel default pagination dengan Bootstrap styling

Card Components:

- `<div class="card">` - Container untuk form dan details
- `<div class="card-header">` - Header card
- `<div class="card-body">` - Body card

Alert Components:

- `<div class="alert alert-warning">` - Amaran duplicate detection
- `<div class="alert alert-success">` - Success message
- `<div class="alert alert-danger">` - Error message

Badge Components:

- `Aktif`
- `Draft`
- `Tidak Aktif`

Modal Components:

- `<div class="modal">` - Modal untuk confirm delete/deactivate
- SweetAlert2 untuk confirmation dialogs (sudah ada dalam project)

Icons

- **Modul Icon:** `heroicon-o-users` (Users icon)
- **SubModul Icon:** `heroicon-o-user-plus` (User add icon)
- **Search Icon:** `heroicon-o-magnifying-glass`
- **Edit Icon:** `heroicon-o-pencil-square`
- **Delete Icon:** `heroicon-o-trash`
- **Approve Icon:** `heroicon-o-check-circle`

Tailwind CSS & Dark Mode

Custom Styles:

- Tiada custom Tailwind classes diperlukan, guna Bootstrap 5 sahaja (ikut existing stack)

Dark Mode Support:

- Tidak diperlukan untuk Phase 1 (CoreUI admin template default sahaja)

5.6 Form Requests & Validation

Form Request: `StorePesakitRequest`

Path: `app/Http/Requests/StorePesakitRequest.php`

Validation Rules:

```

public function rules(): array
{
    return [
        'ic_no' => [
            'required_without:passport_no',
            'nullable',
            'string',
            'size:12',
            'regex:/^[0-9]{12}$/ ',
            'unique:pesakits,ic_no'
        ],
        'passport_no' => [
            'required_without:ic_no',
            'nullable',
            'string',
            'max:20',
            'unique:pesakits,passport_no'
        ],
        'name' => 'required|string|max:255',
        'date_of_birth' => 'required|date|before:today',
        'gender' => 'required|in:Lelaki,Perempuan',
        'phone_primary' => [
            'required',
            'regex:/^01[0-9]-?[0-9]{7,8}$/ ',
            'unique:pesakits,phone_primary'
        ],
        'phone_alternative' => [
            'nullable',
            'regex:/^01[0-9]-?[0-9]{7,8}$/ '
        ],
        'email' => 'nullable|email|max:255|unique:pesakits,email',
        'address' => 'required|string|max:500',
        'postcode' => 'required|string|size:5|regex:/^[0-9]{5}$/ ',
        'city' => 'required|string|max:100',
        'state' => 'required|in:' . implode(',', config('pesakit.states')),
        'blood_type' => 'nullable|in:A+,A-,B+,B-,AB+,AB-,O+,O-',
        'allergies' => 'nullable|string',
        'emergency_contact_name' => 'required|string|max:255',
        'emergency_contact_phone' => [
            'required',
            'regex:/^01[0-9]-?[0-9]{7,8}$/ '
        ],
        'emergency_contact_relationship' => 'nullable|string|max:50',
        'pdpa_consent' => 'required|accepted',
    ];
}

```

Custom Error Messages:

```

public function messages(): array
{
    return [
        'ic_no.required_without' => 'No. IC diperlukan jika tiada No. Passport.',
        'ic_no.size' => 'No. IC mesti 12 digit tanpa tanda.',
        'ic_no.regex' => 'Format No. IC tidak sah (contoh: 991231145678).',
    ];
}

```

```

        'ic_no.unique' => 'No. IC ini sudah wujud dalam sistem.',
        'passport_no.required_without' => 'No. Passport diperlukan jika tiada No. IC.',
        'passport_no.unique' => 'No. Passport ini sudah wujud dalam sistem.',
        'phone_primary.regex' => 'Format No. Telefon tidak sah (contoh: 012-3456789).',
        'phone_primary.unique' => 'No. Telefon ini sudah wujud dalam sistem.',
        'pdpa_consent.accepted' => 'Anda mesti bersetuju dengan Akta Perlindungan Data Peribadi.',
    ];
}

```

Form Request: `UpdatePesakitRequest`

Path: `app/Http/Requests/UpdatePesakitRequest.php`

Validation Rules: (Sama seperti Store, tapi unique exclude current record)

```

public function rules(): array
{
    $pesakitId = $this->route('pesakit')->id;

    return [
        'ic_no' => [
            'required_without:passport_no',
            'nullable',
            'string',
            'size:12',
            'regex:/^[0-9]{12}$/ ',
            "unique:pesakits,ic_no,{ $pesakitId}"
        ],
        // ... same as StorePesakitRequest with unique exceptions
    ];
}

```

6. Logik Perniagaan & Workflow

6.1 Workflow Utama Pengguna

Flow 1: Pendaftaran Pesakit Baharu oleh Kerani

1. Kerani login ke sistem
2. Klik menu "Pesakit" → "Daftar Baharu"
3. Sistem papir form pendaftaran kosong
4. Kerani isi maklumat pesakit:
 - Masukkan No. IC (auto-detect DOB & jantina)
 - Nama penuh
 - No. telefon
 - Alamat, poskod, bandar, negeri
 - Email (optional)
 - Kumpulan darah (optional)
 - Alahan (optional)
 - Maklumat waris terdekat
5. Kerani tick checkbox "Saya bersetuju dengan Akta PDPA"
6. Kerani klik "Simpan sebagai Draft"
7. Sistem validate input:
 - Jika IC sudah wujud: Papar modal amaran
 - "Pesakit dengan IC ini sudah wujud. Guna rekod sedia ada?"

- Pilihan: [Lihat Rekod] [Buat Baru Anyway]
 - Jika IC baru: Proceed
8. Sistem:
 - Generate Patient ID (P2026-0001)
 - Rekod PDPA consent dengan timestamp dan user ID
 - Simpan sebagai status "Draft"
 - Log audit trail (action: create)
 9. Sistem redirect ke halaman detail pesakit
 10. Papar success message: "Pesakit berjaya didaftarkan. Menunggu pengesahan Jururawat."

Flow 2: Pengesahan Pesakit oleh Jururawat

1. Jururawat login ke sistem
2. Klik menu "Pesakit" → "Senarai Pesakit"
3. Sistem papar senarai pesakit dengan filter status "Draft"
4. Jururawat klik "Lihat" pada pesakit yang perlu disahkan
5. Sistem papar detail pesakit lengkap
6. Jururawat semak maklumat:
 - Jika ada kesilapan: Klik "Edit" → Betulkan → Simpan
 - Jika betul: Klik "Sahkan & Aktifkan"
7. Sistem papar modal confirmation
8. Jururawat confirm
9. Sistem:
 - Update status dari "Draft" ke "Aktif"
 - Rekod approved_by dan approved_at
 - Log audit trail (action: approve)
10. Sistem redirect balik ke senarai
11. Papar success message: "Pesakit P2026-0001 telah disahkan dan aktif."

Flow 3: Carian Pesakit

1. Kerani/Jururawat klik menu "Pesakit" → "Carian"
2. Sistem papar form carian dengan medan:
 - Cari by: [Dropdown: IC, Nama, Telefon, ID Pesakit]
 - Kata kunci: [Input field]
3. User pilih kriteria dan taip kata kunci
Contoh: Cari by "IC", kata kunci "991231145678"
4. User klik "Cari"
5. Sistem query database (case-insensitive, LIKE)
6. Sistem papar hasil carian:
 - Jika tiada hasil: Papar "Tiada pesakit dijumpai"
 - Jika ada: Papar senarai dalam table
7. User boleh klik "Lihat" untuk view detail
atau "Edit" untuk kemaskini

Flow 4: Kemaskini Maklumat Pesakit

1. User (Kerani/Jururawat) cari dan buka detail pesakit
2. Klik button "Edit"
3. Sistem papar form dengan data sedia ada pre-filled
4. User ubah maklumat (contoh: alamat, telefon)
5. User klik "Simpan"
6. Sistem validate input
7. Sistem:
 - Rekod old values dan new values dalam audit log

- Update rekod pesakit
 - Log audit trail (action: update)
8. Sistem redirect ke halaman detail
9. Papar success message: "Maklumat pesakit berjaya dikemaskini."

6.2 Pengurusan State

Livewire Properties: (Tidak berkenaan - guna standard Blade)

Controller Properties:

```
// Dalam PesakitController
protected PesakitService $service;
protected PesakitRepository $repository;

// Filter states untuk senarai
$status = $request->get('status', 'all'); // all, Draft, Aktif, Tidak Aktif
$search = $request->get('search', '');
$perPage = 15;
```

6.3 Event Handling

Events Dispatched:

Tidak diperlukan untuk Phase 1 (standard CRUD sahaja).

Untuk future enhancement (Phase 2):

- PesakitCreated - Bila pesakit baharu didaftar
- PesakitApproved - Bila pesakit di-approve oleh Jururawat
- PesakitUpdated - Bila maklumat dikemaskini
- PesakitDeactivated - Bila pesakit ditandakan tidak aktif

6.4 Background Jobs (Tidak Berkenaan)

Tiada background job untuk Phase 1.

Untuk future enhancement:

- Weekly email report kepada Admin (senarai pesakit baharu)
- Auto-deactivate pesakit yang tidak kunjung lebih 2 tahun

7. Keperluan Ujian

7.1 Feature Tests

Path: tests/Feature/Pesakit/PesakitControllerTest.php

```
// Test: User boleh view senarai pesakit
test('kerani can view pesakit list', function () {
    $user = User::factory()->create();
    // Assign role Kerani

    $response = $this->actingAs($user)
        ->get(route('admin.pesakit.index'));

    $response->assertOk();
    $response->assertViewIs('admin.pesakit.index');
```

```

});

// Test: User boleh daftar pesakit baharu
test('kerani can create new pesakit', function () {
    $user = User::factory()->create();

    $data = [
        'ic_no' => '991231145678',
        'name' => 'Ahmad Bin Ali',
        'phone_primary' => '012-3456789',
        'address' => 'No 1, Jalan Test',
        'postcode' => '50000',
        'city' => 'Kuala Lumpur',
        'state' => 'Wilayah Persekutuan',
        'emergency_contact_name' => 'Siti Binti Ahmad',
        'emergency_contact_phone' => '013-9876543',
        'pdpa_consent' => true,
    ];

    $response = $this->actingAs($user)
        ->post(route('admin.pesakit.store'), $data);

    $response->assertRedirect();
    $this->assertDatabaseHas('pesakits', [
        'ic_no' => '991231145678',
        'name' => 'Ahmad Bin Ali',
        'status' => 'Draft',
    ]);
});

// Test: Jururawat boleh approve pesakit
test('jururawat can approve pesakit', function () {
    $jururawat = User::factory()->create();
    // Assign role Jururawat

    $pesakit = Pesakit::factory()->create(['status' => 'Draft']);

    $response = $this->actingAs($jururawat)
        ->patch(route('admin.pesakit.approve', $pesakit));

    $response->assertRedirect();
    $this->assertDatabaseHas('pesakits', [
        'id' => $pesakit->id,
        'status' => 'Aktif',
    ]);
});

// Test: Duplicate IC detection
test('system warns on duplicate ic_no', function () {
    $user = User::factory()->create();
    $existing = Pesakit::factory()->create(['ic_no' => '991231145678']);

    $data = [
        'ic_no' => '991231145678', // Same IC
        'name' => 'Different Person',
        // ... other fields
    ];

```

```

];

$response = $this->actingAs($user)
    ->post(route('admin.pesakit.store'), $data);

$response->assertSessionHasErrors('ic_no');
});

// Test: User boleh cari pesakit by IC
test('user can search pesakit by ic', function () {
    $user = User::factory()->create();
    $pesakit = Pesakit::factory()->create(['ic_no' => '991231145678']);

    $response = $this->actingAs($user)
        ->get(route('admin.pesakit.search', ['search' => '991231145678']));

    $response->assertOk();
    $response->assertSee($pesakit->name);
});

// Test: Unauthorized user tak boleh access
test('guest cannot access pesakit routes', function () {
    $response = $this->get(route('admin.pesakit.index'));
    $response->assertRedirect(route('admin.login'));
});

// Test: Validation rules enforced
test('validation fails if required fields missing', function () {
    $user = User::factory()->create();

    $response = $this->actingAs($user)
        ->post(route('admin.pesakit.store'), []);

    $response->assertSessionHasErrors([
        'name', 'phone_primary', 'address', 'postcode', 'city', 'state'
    ]);
});

// Test: Audit log created on update
test('audit log created when pesakit updated', function () {
    $user = User::factory()->create();
    $pesakit = Pesakit::factory()->create(['phone_primary' => '012-3456789']);

    $this->actingAs($user)
        ->patch(route('admin.pesakit.update', $pesakit), [
            'phone_primary' => '013-9999999',
            // ... other required fields
        ]);

    $this->assertDatabaseHas('pesakit_audit_logs', [
        'pesakit_id' => $pesakit->id,
        'user_id' => $user->id,
        'action' => 'update',
    ]);
});

```

7.2 Unit Tests

Path: tests/Unit/PesakitTest.php

```
// Test: Model relationships
test('pesakit belongs to registered_by user', function () {
    $user = User::factory()->create();
    $pesakit = Pesakit::factory()->create(['registered_by' => $user->id]);

    expect($pesakit->registeredBy)->toBeInstanceOf(User::class);
    expect($pesakit->registeredBy->id)->toBe($user->id);
});

// Test: Auto-generate patient_id
test('patient_id auto-generated on create', function () {
    $pesakit = Pesakit::factory()->create();

    expect($pesakit->patient_id)->toStartWith('P2026-');
    expect(strlen($pesakit->patient_id))->toBe(10);
});

// Test: Auto-extract DOB from IC
test('date_of_birth extracted from ic_no', function () {
    $pesakit = Pesakit::factory()->create(['ic_no' => '991231145678']);

    expect($pesakit->date_of_birth->format('Y-m-d'))->toBe('1999-12-31');
});

// Test: Auto-extract gender from IC
test('gender extracted from ic_no', function () {
    $lelaki = Pesakit::factory()->create(['ic_no' => '991231145679']); // Odd last digit
    $perempuan = Pesakit::factory()->create(['ic_no' => '991231145678']); // Even last digit

    expect($lelaki->gender)->toBe('Lelaki');
    expect($perempuan->gender)->toBe('Perempuan');
});

// Test: Scope aktif
test('scope aktif returns only active pesakits', function () {
    Pesakit::factory()->create(['status' => 'Aktif']);
    Pesakit::factory()->create(['status' => 'Draft']);

    $aktif = Pesakit::aktif()->get();

    expect($aktif)->toHaveCount(1);
    expect($aktif->first()->status)->toBe('Aktif');
});

// Test: Formatted IC accessor
test('formatted_ic accessor adds dashes', function () {
    $pesakit = Pesakit::factory()->create(['ic_no' => '991231145678']);

    expect($pesakit->formatted_ic)->toBe('991231-14-5678');
});

// Test: Age accessor
```

```
test('age accessor calculates correctly', function () {
    $pesakit = Pesakit::factory()->create([
        'date_of_birth' => now()->subYears(25)
    ]);

    expect($pesakit->age)->toBe(25);
});
```

7.3 Data Ujian

Factories:

Path: database/factories/PesakitFactory.php

```
public function definition(): array
{
    $icNo = $this->faker->numerify('#####');
    $lastDigit = rand(0, 9);
    $icNo .= $lastDigit;

    return [
        'ic_no' => $icNo,
        'name' => $this->faker->name(),
        'phone_primary' => '012-' . $this->faker->numerify('#####'),
        'email' => $this->faker->unique()->safeEmail(),
        'address' => $this->faker->address(),
        'postcode' => $this->faker->numerify('#####'),
        'city' => $this->faker->city(),
        'state' => $this->faker->randomElement(config('pesakit.states')),
        'emergency_contact_name' => $this->faker->name(),
        'emergency_contact_phone' => '013-' . $this->faker->numerify('#####'),
        'status' => 'Aktif',
        'pdpa_consent' => true,
        'pdpa_consent_at' => now(),
    ];
}
```

Seeders:

Path: database/seeders/PesakitSeeder.php

```
public function run(): void
{
    // Create 50 sample pesakits
    Pesakit::factory()->count(50)->create();

    // Create 10 draft pesakits
    Pesakit::factory()->count(10)->create(['status' => 'Draft']);
}
```

7.4 Sasaran Coverage Ujian

- ✓ Semua happy paths tested (create, read, update, search, approve)
- ✓ Semua validation rules tested (IC format, unique constraints, required fields)
- ✓ Semua authorization checks tested (guest, kerani, jururawat)

- ☒ Edge cases covered (duplicate IC, warganegara asing, kanak-kanak)
- ☒ Audit trail tested (log created on update/approve)
- ☒ Auto-generation tested (patient_id, DOB/gender from IC)

Target Coverage: > 90%

8. Langkah Implementasi

8.1 Checklist Pra-Implementasi

- ☒ PRD disemak dan diluluskan
- ☐ Design mockups (wireframes untuk form dan senarai) - **Optional untuk Phase 1**
- ☒ Dependencies dikenal pasti (Laravel 12, Bootstrap 5, CoreUI)
- ☒ Database schema disemak

8.2 Urutan Implementasi

Langkah 1: Setup Database

```
# Create migration
php artisan make:migration create_pesakits_table
php artisan make:migration create_pesakit_audit_logs_table

# Create model
php artisan make:model Pesakit
php artisan make:model PesakitAuditLog

# Run migrations
php artisan migrate
```

Langkah 2: Create Configuration File

```
# Create config file manually
touch config/pesakit.php
```

Isi kandungan:

```
<?php
return [
    'states' => [
        'Johor', 'Kedah', 'Kelantan', 'Melaka', 'Negeri Sembilan',
        'Pahang', 'Pulau Pinang', 'Perak', 'Perlis', 'Sabah',
        'Sarawak', 'Selangor', 'Terengganu', 'Wilayah Persekutuan'
    ],
    'blood_types' => ['A+', 'A-', 'B+', 'B-', 'AB+', 'AB-', 'O+', 'O-'],
];
```

Langkah 3: Generate FormRequests

```
php artisan make:request StorePesakitRequest
php artisan make:request UpdatePesakitRequest
```

Implement validation rules seperti dalam Section 5.6.

Langkah 4: Create Service & Repository

```
mkdir app/Services
mkdir app/Repositories

# Create files manually
touch app/Services/PesakitService.php
touch app/Repositories/PesakitRepository.php
```

Langkah 5: Implement Business Logic

- Update `Pesakit` model:
 - Relationships
 - Scopes (aktif, draft)
 - Accessors (formatted_ic, age, status_badge)
 - Boot method (auto-generate patient_id, extract DOB/gender from IC)
- Implement `PesakitService` :
 - CRUD operations
 - Duplicate detection logic
 - Approval logic
- Implement `PesakitRepository` :
 - Database queries
 - Search functionality

Langkah 6: Update Controller

Update `app/Http/Controllers/Admin/PesakitController.php` :

- Inject `PesakitService`
- Implement all CRUD methods
- Implement search method
- Implement approve method
- Add audit logging

Langkah 7: Build UI (Blade Views)

Create views:

```
resources/views/admin/pesakit/
├─ index.blade.php      (Senarai pesakit dengan pagination)
├─ create.blade.php     (Form pendaftaran)
├─ edit.blade.php       (Form kemaskini)
├─ show.blade.php       (Detail pesakit)
└─ search.blade.php     (Form carian)
```

Guna Bootstrap 5 + CoreUI components.

Langkah 8: Audit Trail Implementation

- Create trait `HasAuditTrail` untuk auto-log perubahan
- Attach trait to `Pesakit` model
- Create observer untuk capture old/new values

Langkah 9: Add to Navigation

Update sidebar menu:

```
// resources/views/components/admin-sidebar.blade.php

<li class="nav-item">
  <a class="nav-link" href="{{ route('admin.pesakit.index') }}">
    <i class="icon-user nav-icon"></i> Pesakit
  </a>
</li>
```

Langkah 10: Write Tests

```
# Create test files
php artisan make:test Pesakit/PesakitControllerTest
php artisan make:test Pesakit/PesakitTest --unit

# Create factories
php artisan make:factory PesakitFactory
php artisan make:factory PesakitAuditLogFactory

# Run tests
php artisan test --filter=Pesakit
```

Langkah 11: Code Formatting

```
vendor/bin/pint --dirty
```

Langkah 12: Final Review

- ☐ Semua tests passing
- ☐ Code formatted dengan Pint
- ☐ Audit trail working
- ☐ Duplicate detection working
- ☐ PDPA consent recorded
- ☐ Navigation menu updated
- ☐ PRD acceptance criteria dipenuhi

9. Dependencies

9.1 Pakej Luaran

Tiada pakej tambahan diperlukan. Semua menggunakan Laravel 12 standard dan library sedia ada:

- **spatie/laravel-route-attributes** (sudah ada) - Untuk Route Attributes
- **Bootstrap 5** (sudah ada) - UI framework
- **CoreUI** (sudah ada) - Admin template
- **SweetAlert2** (sudah ada) - Confirmation dialogs

9.2 Ciri/Modul Berkaitan

Bergantung Kepada:

- User Authentication (sudah wujud)
- Role management (perlu pastikan role Kerani dan Jururawat wujud)

Memberi Kesan Kepada (Future):

- Modul Temujanji (akan guna patient_id sebagai foreign key)
- Modul EMR (akan guna patient_id sebagai foreign key)
- Modul Billing (akan guna patient_id sebagai foreign key)

Catatan: Database schema untuk `pesakit` table perlu stabil kerana modul lain akan bergantung kepadanya.

9.3 Integrasi Pihak Ketiga

Tiada untuk Phase 1.

Future enhancement (Phase 2):

- MyKad Reader SDK/API (jika IC scanner diperlukan)
- SMS Gateway (untuk notifikasi temujanji kepada pesakit)
- Email service (untuk send PDPA consent form)

10. Kriteria Penerimaan

10.1 Penerimaan Fungsi

- ☒ **FR-01 hingga FR-14** dilaksanakan sepenuhnya
- ☒ Semua user stories (US-01 hingga US-10) boleh diselesaikan dengan jayanya
- ☒ Kebenaran dan authorization berfungsi seperti yang dinyatakan (Kerani vs Jururawat)
- ☒ Sistem badge tidak berkenaan (N/A)
- ☒ Data validation menguatkuasakan semua peraturan (IC format, unique constraints, required fields)
- ☒ Error handling memberikan maklumbalas yang jelas (SweetAlert2 alerts, validation errors)

10.2 Penerimaan Teknikal

- ☒ Semua feature tests lulus (minimum 15 test cases)
- ☒ Semua unit tests lulus (minimum 10 test cases)
- ☒ Code mengikuti konvensyen Laravel dari `.github/copilot-instructions.md` :
 - ☒ Route Attributes pattern
 - ☒ Repository + Service pattern
 - ☒ FormRequests untuk validation
 - ☒ HandlesApiResponses trait untuk error handling
 - ☒ Audit logging untuk semua mutations
- ☒ Code formatted dengan `vendor/bin/pint`
- ☒ Tiada N+1 query problems (eager loading digunakan untuk relationships)
- ☒ Blade component mempunyai single root element (standard Bootstrap structure)
- ☒ Dark mode support tidak diperlukan untuk Phase 1

10.3 Penerimaan Kualiti

- ☒ Code disemak oleh peer (code review)
- ☒ Manual testing selesai (test semua workflows: daftar, approve, search, update)
- ☒ Tiada console errors atau warnings
- ☒ Responsive design berfungsi pada mobile/tablet (Bootstrap 5 responsive)
- ☒ Pertimbangan accessibility (labels untuk form fields, ARIA attributes)

10.4 Penerimaan Dokumentasi

- ☒ PRD dikemaskini dengan nota implementasi akhir
- ☐ BADGES.md tidak berkenaan (tiada badge notification)
- ☐ README.md tidak perlu dikemaskini (internal admin feature)

- ☒ Inline code comments untuk logik kompleks (contoh: IC parsing logic, patient_id generation)
-

11. Nota & Pertimbangan

11.1 Peningkatan Masa Hadapan

Phase 2 Enhancements:

1. IC Scanner Integration

- Hardware scanner support (APAD SmartD, Dermalog)
- OCR untuk auto-fill form dari gambar IC
- Reduce manual data entry time

2. Advanced Search & Filters

- Fuzzy search untuk nama (typo tolerance)
- Advanced filters (tarikh daftar, kumpulan darah, status)
- Export senarai filtered ke Excel/PDF

3. Bulk Operations

- Bulk import pesakit dari Excel
- Bulk update (contoh: update negeri untuk semua pesakit)
- Bulk deactivate

4. Integration dengan Modul Lain

- Auto-link pesakit dengan temujanji
- Auto-populate EMR dengan data pesakit
- Auto-generate invoice dalam Billing

5. Patient Portal (Self-service)

- Pesakit boleh login dan update maklumat sendiri
- Pesakit boleh request appointment online
- Pesakit boleh view medical records

6. SMS/Email Notifications

- Welcome SMS/email selepas pendaftaran
- Reminder untuk temujanji
- Birthday wishes automation

7. Analytics & Reporting

- Dashboard: Jumlah pesakit baharu per bulan
- Demographics report (umur, jantina, lokasi)
- Retention report (pesakit aktif vs tidak aktif)

11.2 Batasan Yang Diketahui

1. Tiada IC Scanner untuk Phase 1

- Manual data entry sahaja
- Future: Integrate hardware scanner atau OCR

2. Duplicate Detection Simple

- Check by IC/Passport sahaja
- Tiada fuzzy matching untuk nama atau DOB
- Future: Implement fuzzy logic untuk catch typo IC

3. Tiada Integration dengan Modul Lain

- Standalone module untuk Phase 1
- Future: API endpoints untuk Temujanji, EMR, Billing

4. Audit Trail Basic

- Log perubahan sahaja (old/new values)
- Tiada detailed user activity tracking
- Future: Full activity log dengan IP, browser fingerprint

5. Tiada Export/Import

- Manual entry sahaja untuk Phase 1
- Future: Bulk import dari Excel, export ke PDF/Excel

11.3 Pertimbangan Migrasi

Catatan:

- Ini adalah modul baharu (tiada data lama untuk migrate)
- Jika klinik ada sistem lama, perlu plan untuk:
 - Data mapping (format IC lama vs baharu)
 - Bulk import script
 - Data cleaning (remove duplicates, fix format)

11.4 Pertimbangan Prestasi

1. Database Indexing

- Index pada `ic_no`, `name`, `phone_primary` untuk carian pantas
- Composite index untuk search queries

2. Pagination

- Default 15 rekod per page (sesuai untuk admin)
- Lazy loading jika data bertambah (> 10,000 pesakit)

3. Caching (Optional untuk Phase 2)





- Cache dropdown lists (negeri, kumpulan darah)
- Cache statistics untuk dashboard

4. Query Optimization




- Eager load relationships untuk avoid N+1
- Use `select()` untuk fetch specific columns sahaja

11.5 Pertimbangan Keselamatan

1. PDPA Compliance

-  Consent management (mandatory checkbox)
-  Soft delete (data retention 10 tahun)
-  Audit trail (track who access/change data)
-  Data encryption at rest (Future: encrypt IC, phone, email)

2. Access Control

-  Role-based permissions (Kerani vs Jururawat)
-  Auth middleware pada semua routes
-  Row-level security (Future: Kerani hanya edit pesakit sendiri)

3. Input Validation

- ☒ Server-side validation (FormRequest)
- ☒ Client-side validation (HTML5 + Bootstrap)
- ☒ SQL injection prevention (Eloquent ORM)
- ☒ XSS prevention (Blade auto-escape)

4. Data Privacy

- ☒ IC/Passport unique (prevent duplicate)
- ☐ Masking IC (Future: Show 991231-XX-XXXX in list view)
- ☐ Data encryption (Future: Encrypt sensitive fields)

12. Appendix

12.1 Rujukan

- **PRD Berkaitan:**
 - Tiada (ini adalah modul pertama untuk sistem klinik)
- **Design Mockups:**
 - Tiada untuk Phase 1 (wireframes simple sahaja)
- **API Documentation:**
 - Tidak berkenaan (tiada API untuk Phase 1)
- **Laravel Documentation:**
 - [Laravel 12 Validation](#)
 - [Eloquent Relationships](#)
 - [Route Model Binding](#)
- **Spatie Route Attributes:**
 - [Documentation](#)

12.2 Change Log

Tarikh	Pengarang	Perubahan
12 Jan 2026	Pasukan Pembangunan	PRD awal dicipta berdasarkan keperluan pengguna

12.3 Kelulusan

- ☐ **Pemilik Produk:** [Nama Pemilik Klinik] - [Tarikh]
- ☐ **Ketua Teknikal:** [Nama] - [Tarikh]
- ☐ **Stakeholders:** [Nama Kerani, Jururawat] - [Tarikh]

Status Implementasi: Belum Bermula **Tarikh Siap Dijangka:** [Tarikh bila feature ready untuk production] **Tarikh Selesai:** [Tarikh bila feature deployed ke production]

Lampiran A: Contoh Data Pesakit

Contoh 1: Pesakit Warganegara Malaysia

```
{
  "patient_id": "P2026-0001",
  "ic_no": "991231145678",
```

```
{
  "name": "Ahmad Bin Ali",
  "date_of_birth": "1999-12-31",
  "gender": "Lelaki",
  "phone_primary": "012-3456789",
  "email": "ahmad@example.com",
  "address": "No 123, Jalan Bunga Raya",
  "postcode": "50000",
  "city": "Kuala Lumpur",
  "state": "Wilayah Persekutuan",
  "blood_type": "O+",
  "allergies": "Penisilin",
  "emergency_contact_name": "Siti Binti Ahmad",
  "emergency_contact_phone": "013-9876543",
  "emergency_contact_relationship": "Isteri",
  "status": "Aktif",
  "pdpa_consent": true,
  "pdpa_consent_at": "2026-01-12 09:30:00"
}
```

Contoh 2: Pesakit Warganegara Asing

```
{
  "patient_id": "P2026-0002",
  "passport_no": "A12345678",
  "name": "John Doe",
  "date_of_birth": "1985-05-15",
  "gender": "Lelaki",
  "phone_primary": "012-9999999",
  "email": "john@example.com",
  "address": "Unit 5-10, Condominium ABC",
  "postcode": "50450",
  "city": "Kuala Lumpur",
  "state": "Wilayah Persekutuan",
  "country": "Singapore",
  "blood_type": "A+",
  "emergency_contact_name": "Jane Doe",
  "emergency_contact_phone": "012-8888888",
  "emergency_contact_relationship": "Wife",
  "status": "Aktif",
  "pdpa_consent": true
}
```

Contoh 3: Pesakit Kanak-kanak

```
{
  "patient_id": "P2026-0003",
  "ic_no": null,
  "passport_no": null,
  "name": "Ali Bin Ahmad",
  "date_of_birth": "2020-03-10",
  "gender": "Lelaki",
  "phone_primary": "012-3456789",
  "address": "No 123, Jalan Bunga Raya",
  "postcode": "50000",
}
```

```
"city": "Kuala Lumpur",  
"state": "Wilayah Persekutuan",  
"emergency_contact_name": "Ahmad Bin Ali (Bapa)",  
"emergency_contact_phone": "012-3456789",  
"emergency_contact_relationship": "Bapa",  
"status": "Aktif",  
"pdpa_consent": true,  
"pdpa_consent_at": "2026-01-12 10:00:00"  
}
```

TAMAT PRD