HMS — Healthcare & Medical Services Management

Detailed Requirements Specification (English only)
Generated by ChatGPT
Date: 2025-09-27

Revision History

Version 1.0 — Detailed requirements (initial).

Author: Generated by ChatGPT (requirements level, user requested 'perfect' detailed spec).

Table of Contents

1. Project Overview and Scope

Purpose:
HMS is a domain-focused requirements specification for a clinic/hospital management backend system. It emphasizes complex business logic, data integrity, validation, and realistic healthcare workflows — not just UI screens. The goal is to provide a detailed logical specification you can implement in any stack.

Scope:
- Patient lifecycle management (registration, identification, status)

- Scheduling and appointment conflict resolution
- Clinical visits and documentation
- Prescriptions and pharmacy fulfillment (including drug-interaction checks)
- Test orders, results, and pre-condition checks
- Billing, payments, and insurance claims lifecycle
- User roles, permissions, audit, and system administration

Out of scope:
- Specific UI mockups or technology stacks
- Integration with external third-party providers (unless specified in later iterations)

2. Actors and Roles

- System Administrator — full admin privileges; manages users, roles, and system configuration.
- Receptionist — registers patients, schedules appointments, accepts payments, creates simple visits.
- Doctor — views patient history, performs visits, prescribes medications, orders tests.
- Nurse — assists clinical staff; can view patient records and update vitals and notes.
- Lab Technician — receives test orders, records results and status.
- Pharmacist — reviews prescriptions and records dispensing transactions.
- Billing Officer — reviews bills, records payments, submits insurance claims.
- Insurance Agent (external actor) — reviews claims and returns approvals/rejections.
- Patient (portal) — optional actor who can view appointments and bills.

3. Glossary of Terms

Visit: A completed clinical interaction tied to an Appointment and containing diagnosis, prescriptions, tests, and billable items.

Appointment: A scheduled time slot between a patient and a doctor. Becomes a Visit when completed.

Claim: An insurance claim generated for services rendered during a visit.

Dispense: Pharmacy action to provide medication to a patient against a prescription.

TestOrder: A request from a clinician to perform a laboratory or imaging test.

4. High-level Services / Features

22. Patient Registration & Unique Identity Management
23. Doctor & Department Management (schedules and specialties)
24. Appointment Scheduling (conflict detection and cancellation policies)
25. Clinical Visit Management (notes, diagnoses, visit closure rules)
26. Prescriptions & Pharmacy Fulfillment (interaction checks, dispensing)
27. Medical Test Orders & Lab Results (pre-conditions enforcement)

28. Billing, Payments & Refunds (per-visit billing, itemization)
29. Insurance Claim Lifecycle (create, submit, approve/reject, settle)
30. User Management, Permissions & Audit Logs
31. Reporting and Notifications (emails, SMS, reminders)

5. Detailed Module Requirements

5.1 Patient Management

Purpose: Maintain a single source of truth for patient identity and demographics.

Actors: Receptionist, Patient (self-service), Doctor, Admin

User Stories:

32. As a Receptionist, I can create a new patient with National ID, name, DOB, contact, and insurance so that future visits are linked to a stable identity.
33. As a Receptionist, I can search for an existing patient by National ID, name, phone, or email to avoid duplicates.
34. As a Doctor, I can view a patient's complete demographic and visit history before a consultation.
35. As a Patient, I can update my contact information through a portal (subject to verification).

Data Fields (minimum):

- PatientID (PK) — system generated
- NationalID — unique, required, validated format per country
- FullName
- DateOfBirth
- Gender
- Address
- Phone
- Email
- EmergencyContact (name, relation, phone)
- InsuranceProviderID (nullable)
- Status (Active, Inactive, Deceased)
- Allergies (free text or linked allergy table)
- ChronicConditions (linked table or tags)

Business Rules & Validations:

- NationalID must be unique. Duplicate NationalID blocks creation.
- Age must be computable from DateOfBirth; system should validate reasonable ages (e.g., 0 < age < 130).
- Mark patient Inactive after 3 no-shows (configurable).

- Deceased patients cannot have new appointments created.
- Contact info changes should be timestamped and auditable.

Workflows / States:

Patient status transitions: Active → Inactive → Reactivated (manual) or Deceased (blocked).

5.2 Doctor & Department Management

Purpose: Manage clinical staff, their specialties, licenses, and availability.

User Stories:

36. As an Admin, I can add a doctor profile including license number, specialties, and an availability schedule.
37. As an Admin, I can mark a doctor on leave for a period; the system cancels or reschedules appointments within that period.
38. As a Receptionist, I can view only active doctors when scheduling.

Data Fields (minimum):

- DoctorID (PK)
- FullName
- LicenseNumber (unique)
- Specialization (one or more)
- PrimaryDepartmentID
- AvailabilitySchedule (recurring slots and exceptions)
- Status (Active, OnLeave, Retired)

Business Rules & Validations:

- Doctor license number must be unique and validated for proper format.
- Doctors cannot have overlapping availability ranges defined for themselves.
- On setting OnLeave status, existing appointments are flagged and receptionist notified to reschedule or cancel.
- Certain services are restricted to specific specializations (e.g., cardiology procedures).

5.3 Appointment Scheduling

Purpose: Provide robust scheduling with conflict resolution, cancellation policies, and appointment states.

Actors: Receptionist, Patient, Doctor

Key User Stories:

39. As a Receptionist, I can create an appointment specifying patient, doctor, date, start time, and end time.

40. As a Receptionist, I want the system to prevent booking conflicting appointments for the same doctor.
41. As a Patient, I receive an appointment reminder 48 and 24 hours before the appointment.
42. As a Receptionist, I can cancel or reschedule an appointment; if canceled less than 24 hours before, a cancellation fee applies.

Appointment Data Fields:

- AppointmentID (PK)
- PatientID (FK)
- DoctorID (FK)
- DepartmentID (FK)
- Date, StartTime, EndTime
- Status (New, Confirmed, Checked-in, No-Show, Cancelled, Completed)
- Source (Walk-in, Portal, Phone, Referral)
- Fees (computed from service/doctor)
- Notes

Business Rules & Validations:

- No two confirmed/checked-in appointments for a doctor may overlap (strict).
- Patient cannot have two active appointments that overlap in time for different doctors.
- Appointment cannot be scheduled in the past.
- Cancellation fee applies if cancellation occurs less than cancellationWindow (configurable, default 24 hours).
- Check-in process changes status to Checked-in and records timestamp; if patient does not check-in within grace period (e.g., 15 minutes), mark as No-Show.

Appointment Lifecycle Example:

New → Confirmed → Checked-in → Completed
New → Confirmed → Cancelled
Confirmed → No-Show

5.4 Medical Visit & Clinical Notes

Purpose: Capture clinical data from consultations and tie clinical decisions to billable events.

User Stories:

43. As a Doctor, I can start a visit from a confirmed appointment and record vital signs, diagnosis, and notes.
44. As a Doctor, I can attach test orders and prescriptions to a visit.
45. As a Nurse, I can record vitals and preliminary observations associated with the visit.

Visit Data Fields:

- VisitID (PK)
- AppointmentID (FK)
- PatientID (FK)
- DoctorID (FK)
- StartTimestamp, EndTimestamp
- Vitals (BP, HR, Temp, SpO2 etc.)
- DiagnosisCodes (ICD or free text)
- ClinicalNotes (long text)
- Prescriptions (linked)
- TestOrders (linked)
- BillingStatus (Open, Billed, Paid)

Business Rules & Validations:

- A Visit can only be created from a Confirmed appointment or as an unscheduled visit (walk-in) with receptionist approval.
- Visit cannot be completed while any mandatory test ordered by the clinician is still in Pending state (configurable).
- Clinical notes are immutable once signed by the doctor; unsigned notes may be edited for a limited time.
- Prescriptions and test orders created during the visit are billable items and should feed into billing automatically.

5.5 Prescriptions & Pharmacy

Purpose: Manage medication prescribing, validate interactions/allergies, and track dispensing.

User Stories:

46. As a Doctor, I can add medications to a prescription with dosage, frequency, duration and notes.
47. As a Pharmacist, I can view pending prescriptions, verify them, and mark them as Dispensed.
48. As a System, I warn prescribers if prescribed medication conflicts with known allergies or interacting drugs.

Prescription Data Fields:

- PrescriptionID (PK)
- VisitID (FK)
- PatientID (FK)
- PrescribedByDoctorID (FK)
- Medications (drug id, name, dose, frequency, duration, instructions)

- Status (Draft, Active, Dispensed, Cancelled)
- CreatedAt, SignedAt

Business Rules & Validations:

- System must check for patient allergies and flag any medication containing an allergen.
- Drug–drug interaction checks should flag potentially dangerous pairs; prescriber must explicitly override with justification if proceeding.
- Prescription must be signed by a licensed doctor before pharmacy can dispense.
- Partial dispensing allowed only if explicitly supported and quantity recorded.
- Controlled substances may require extra authorization and logging.

5.6 Medical Tests & Laboratory

Purpose: Order, track, and record laboratory/imaging tests with pre-conditions and result management.

User Stories:

49. As a Doctor, I can order tests specifying collected samples and any pre-test conditions (e.g., fasting).
50. As a Lab Technician, I can claim a test order, perform the test, and record results and reference ranges.
51. As a Doctor, I can review test results and add interpretation notes.

Test Order Data Fields:

- TestOrderID (PK)
- VisitID (FK)
- TestTypeID (FK)
- OrderedByDoctorID (FK)
- RequestedDate, SampleCollectedDate, CompletedDate
- Status (Ordered, SampleCollected, InProgress, Completed, Cancelled)
- Result (structured or free text), ReferenceRanges, Attachments (images/reports)
- PreConditions (e.g., fasting required)

Business Rules & Validations:

- If a test requires a pre-condition (fasting), system must store that requirement and surface it to patient and staff.
- A TestOrder cannot be marked Completed without a CompletedDate and a Result.
- If sample is not collected within a configurable time window, the test order should be flagged for follow-up or cancellation.
- Certain test types may require a specific technician role to complete.

5.7 Billing and Payments

Purpose: Ensure every billable action is captured, calculated, and paid before closing visits as required.

User Stories:

52. As a System, I compute billable items automatically from appointments, tests, and prescriptions.
53. As a Billing Officer, I can generate an invoice for a visit showing itemized services, insurance contribution, and patient due amount.
54. As a Receptionist, I can accept payments (cash/card) and record them against invoices.

Billing Data Fields:

- InvoiceID (PK)
- VisitID (FK)
- LineItems (service code, description, price, quantity)
- Subtotal, Discounts, InsuranceCoverageAmount, PatientDue, Tax, Total
- Status (Draft, Issued, PartiallyPaid, Paid, Refunded)
- PaymentRecords (date, amount, method, reference)

Business Rules & Validations:

- Invoice total equals sum of line items adjusted by discounts and taxes.
- If patient has insurance, apply coverage rules defined by InsuranceProvider and policy; if policy does not cover a service, it is billed to patient.
- Visit should not be marked Closed if Invoice status is not Paid or if outstanding balance exists unless admin override is applied.
- Refunds must be linked to original payment and recorded separately.

5.8 Insurance Claims

Purpose: Manage claim lifecycle: creation, submission, adjudication, and settlement.

User Stories:

55. As a Billing Officer, I can create a claim from an invoice for insured patients.
56. As a Billing Officer, I can submit a claim to the insurer and change claim status based on responses.
57. As a System, I can apply insurer-approved amounts to patient billing and hospital receivables.

Claim Data Fields:

- ClaimID (PK)
- InvoiceID (FK)
- InsuranceProviderID (FK)

- ClaimStatus (New, Submitted, Approved, Rejected, PartiallyApproved, Settled)
- AmountClaimed, AmountApproved, SubmitDate, ResponseDate, ReferenceNumber
- DenialReasons (if any)

Business Rules & Validations:

- Claims must reference a completed Visit and an Issued Invoice.
- Claims cannot be submitted for invoices already paid in full by patient, unless insurer reimbursement is required.
- Rejected claims must populate denial reason and revert coverage amounts affecting patient due.
- Partial approvals reduce the insurer contribution and increase patient liability accordingly.

5.9 User Management, Security & Audit

Purpose: Manage users, roles, and record system actions for accountability.

Key Requirements:

- Role-based access control (RBAC) with roles: Admin, Receptionist, Doctor, Nurse, LabTech, Pharmacist, BillingOfficer.
- Password policies (complexity, expiration) and account lockout after repeated failed attempts.
- Two-factor authentication support (optional).
- Audit log capturing user, action, timestamp, entity affected, and before/after values for critical changes.

5.10 Reporting & Notifications

Purpose: Provide operational reports and automated notifications to patients and staff.

Reports (examples):

- Daily appointments per doctor
- No-shows and cancellations
- Unpaid invoices and aging report
- Pending test orders older than X days
- Medication dispensing log

Notifications: Email/SMS reminders, lab result notifications, claim status updates. Notification templates configurable.

6. Data Model Overview (entities & key fields)

- Patients (PatientID, NationalID, FullName, DOB, Contact...)
- Doctors (DoctorID, FullName, LicenseNumber, Specialties...)
- Departments (DepartmentID, Name, Description)

- Appointments (AppointmentID, PatientID, DoctorID, Date, StartTime, EndTime, Status)
- Visits (VisitID, AppointmentID, Diagnosis, Notes, BillingStatus)
- Prescriptions (PrescriptionID, VisitID, Status, Medications)
- TestOrders (TestOrderID, VisitID, TestTypeID, Status, Result)
- Invoices (InvoiceID, VisitID, LineItems, Total, Status)
- Claims (ClaimID, InvoiceID, ProviderID, Status)
- Users (UserID, Username, Role, LinkedPersonID)
- AuditLogs (LogID, UserID, Action, Entity, Changes, Timestamp)

Key Relationships and Constraints:

- Patients 1..* Visits
- Doctors 1..* Visits
- Visits 1..1 Appointment (nullable for walk-ins)
- Visit 1..* Prescriptions
- Visit 1..* TestOrders
- Visit 1..1 Invoice (or multiple invoices depending on policy)
- Invoice 0..1 Claim

7. Key Workflows (states & transitions)

Appointment workflow:
New -> Confirmed -> Checked-in -> Completed
New -> Confirmed -> Cancelled
Confirmed -> No-Show

Visit workflow:
New -> InProgress -> Completed -> Billed -> Paid

Prescription workflow:
Draft -> Active (signed) -> Dispensed -> Completed or Cancelled

Claim workflow:
New -> Submitted -> Approved/Rejected -> Settled

Test Order workflow:
Ordered -> SampleCollected -> InProgress -> Completed -> Reported

8. Business Rules and Validation Summary

- Unique patient identity enforced by NationalID.
- Strict no-overlap rule for doctor appointments.
- Visit cannot close with pending mandatory tests.
- Prescriptions must be signed before dispensing.
- Insurance claim lifecycle enforces invoice/visit completion.

- Audit trail required for critical actions (payment, prescription signing, claim adjudication).

9. Iterations and Milestones (detailed)

Suggested 8 iterations to deliver a robust, testable system. Each iteration includes features, acceptance criteria, and testing notes.

58. Iteration 0 — Project Setup & Data Model

- Create project skeleton, database schema base (Patients, Doctors, Departments, Users).
- Acceptance: Can create/read/update/delete patients and doctors with validations.

59. Iteration 1 — User Management & Security

- Implement RBAC, user CRUD, login, password policy, and audit logging.
- Acceptance: Admin can add users and restrict actions by role.

60. Iteration 2 — Appointment Scheduling

- Create appointment flows, conflict detection, cancellation policy, reminders.
- Acceptance: System rejects overlapping doctor appointments; reminders scheduled.

61. Iteration 3 — Visits & Clinical Notes

- Start/complete visits, record vitals and notes, link to appointments.
- Acceptance: Visit cannot be completed with pending mandatory tests (configurable).

62. Iteration 4 — Prescriptions & Pharmacy

- Prescription creation, drug interaction checks, pharmacy dispense logging.
- Acceptance: Prescription must be signed before dispense; interactions trigger warnings.

63. Iteration 5 — Tests & Lab Results

- Order tests, track sample collection and results, attach reports.
- Acceptance: Test cannot complete without result and completion date; pre-conditions enforced.

64. Iteration 6 — Billing & Payments

- Auto-generate invoices, apply insurance coverage rules, accept payments.
- Acceptance: Invoice totals correct; visit closure rules enforced.

65. Iteration 7 — Insurance Claims & Reporting

- Claims lifecycle, reports, notifications, and final polish.
- Acceptance: Claims progress through states and affect accounting correctly.

10. Non-functional Requirements

- Security: encrypt PII at rest, role-based access, secure password handling.
- Performance: support concurrent scheduling operations with low latency (target configurable).
- Reliability: backups, integrity checks, and audit trails.
- Scalability: modular design so modules can be separated or scaled independently.
- Data Retention & Privacy: configurable retention policies, GDPR-like data handling rules.
- Internationalization: store dates/times in UTC, localize format for display.

11. Appendix: User story examples, error messages, acceptance criteria

Sample User Story with Acceptance Criteria:

User Story: As a Receptionist, I can create an appointment and the system will validate doctor availability and prevent conflicts.

Acceptance Criteria:
- Given Doctor D has an appointment at 10:00-10:30, when receptionist attempts to book D for 10:15-10:45, the system rejects the booking and returns conflict information (conflicting appointment id and patient).
- When booking succeeds, an appointment confirmation is recorded and an SMS reminder is scheduled 48 and 24 hours before the appointment.

Sample Error messages and codes (examples):

- ERR-001: Duplicate NationalID — 'A patient with this National ID already exists.'
- ERR-002: AppointmentConflict — 'Requested time conflicts with existing appointment.'
- ERR-003: UnauthorizedAction — 'User does not have permission to perform this action.'
- ERR-004: PendingTests — 'Cannot complete visit: pending mandatory tests exist.'