IT Project 700 Project 2025 – Phase 3
Project Proposal: Doctor Appointment & Office Management System (Enhanced)

Group Members

Member No.	Name	Surname	Student Number	Role
1	LUCIAN MAURICE	Sans-Souci	402306266	Team Leader
2	RICHARD ODISANG	NTWAYAGAE	402100270	Member
3	MVUSULUDZO	NETSHIRANDO	402414431	Member
4	Mitchell	SUNKRAN	402000565	Member
5	MOINUDDEEN	WAHAB	402308091	Member
6	MAHLATSE HEZEKIEL	MADISHA	402307279	Member

Contents

Introduction

Planning phase for “Doctor Appointment & Office Management System (Enhanced)”

This builds directly on our Phase 1 and is tailored to clear up the problem framing, feasibility, rigorous SRS and an assessable plan with PERT/Gantt. It assumes our chosen stack: Python for the backend, HTML/CSS/JS for the frontend, SQLite and MySQL for the Databases, Agile with 6 sprints for each member to lead.

2.1 Identification of need

- Context: Small and rural clinics face high no-show rates, double bookings, long queues, and fragmented tools that are either too expensive or over-engineered for their needs in South Africa.
- Primary stakeholders will be Doctors, receptionists, clinic managers, patients and IT admins.

Core problems:

- Access: Patients struggle to book via preferred channels like phone and emails.
- Reliability: Unstable connectivity breaks workflows and running out on Airtime
- Visibility: No transparent queue/wait times; limited analytics.
- Admin load: Manual reminders, chaotic reschedules, dispersed records.

Objectives:

- Reduce no-shows: Automated reminders + easy cancel/reschedule.
- Streamline ops: Unified calendar, queue, basic records.
- Improve communication: Multi-channel messaging and status updates.
- Enable decisions: Busiest hours, no-show rates, utilization.

Success metrics (target):

- No-shows: −25% within 3 months of deployment.
- Admin time: −30% on booking-related tasks.
- Patient satisfaction: +20% improvement in post-visit rating.
- System uptime: ≥ 99% (with offline tolerance at reception).

Constraints:

- Regulatory: POPIA-aligned data minimization, consent, and access control.
- Budget/time: Student team, 6 sprints, limited infra.
- Skills: Python-centric team; simple web UI.

2.2 Preliminary investigation

Current practice baseline:

- Manual phones and paper diaries, ad-hoc SMS, no analytics, no triage, walk-ins overwhelm schedules.
- Existing solutions (high level):
- Enterprise products exist but are costly, require continuous internet, and don't prioritize WhatsApp/SMS flows or offline-first clinics.
- Proposed solution (high level):
- Python backend with REST APIs.
- Lightweight web UI for reception/admin; patient-facing booking via mobile-friendly web and message links.
- Notifications via SMS/WhatsApp; opt-in email.
- Offline-first reception module: Local cache with sync queue.
- Minimal EMR-lite: Basic patient profile and visit notes (no diagnostics repository).

Key risks:

- Messaging integration complexity, data privacy (POPIA), sync conflicts from offline to online, change resistance at clinics.

Mitigations:

- Scoped MVP messaging (SMS first, WhatsApp Business later), role-based access + audit logs, conflict resolution rules during sync, guided onboarding and import tools.

Existing solutions

Practice Perfect

Description: A comprehensive practice management offering appointment scheduling, SMS reminders, clinical records and billing functions.

Negative: Too complex and expensive for small or rural clinics. Requires staff training and continuous internet connection.

Solution: Our platform focuses only on core functions: booking, reminders, queue, and analytics. It avoids unnecessary complexity and is designed to be lightweight, easy to use, and affordable for smaller clinics.

Navitas

Description: A cloud-based practice management solution with scheduling, reminders, billing, and patient portals.

Negative: Works best with stable connectivity and assumes clinics can always be online. Limited in supporting WhatsApp booking or offline sync.

Solution: Our system includes offline first reception with local caching and sync once internet is restored. It also expands booking channels to WhatsApp and SMS, not just web, making it more accessible to patients in low-resource areas.

Medapp

Description: Medapp offers a simple booking calendar, with SMS/email reminders and patient search by practitioner.

Negative: Lightweight but lacks advanced features such as queue management, urgency flagging and analytics. No offline operation.

Solution: We enhance this simplicity with queue management, triage questionnaires, and dashboards that track no-shows and peak hours, while still keeping the user interface lightweight.

2.3 Feasibility Study

Technical Feasibility

Description:
The project will use a Python backend (Flask/FastAPI), SQLite for offline cache, MySQL for production, and a lightweight HTML/CSS/JavaScript frontend. Integration with SMS/email gateways will handle notifications.

Challenges:

- Offline/online sync may create conflicts.
- SMS/WhatsApp integration can be technically complex.
- Ensuring data security and POPIA compliance (encryption, access control).

Solution:

- Implement “first come first serve” conflict resolution with audit logs.
- Start with SMS only (lower complexity), integrate WhatsApp Business later.
- TLS and hashed passwords to protect patient data.

Operational Feasibility

Description:
The system must fit the day-to-day workflows of small and rural clinics. Receptionists will use it to manage bookings/queues, while doctors access schedules and notes.

Challenges:

- Resistance to change from staff used to paper-based systems.
- Training needs for reception and admin staff.
- Dependence on clinic staff following proper digital processes.

Solution:

- Provide short training sessions (1–2 hours) with simple guides.
- Keep UI lightweight: most common tasks done in +- 3 clicks.
- Pilot in one clinic with strong support and onboarding before wider rollout.

Economic Feasibility

Description:
The project is student-built with low upfront development costs. Cloud costs will be minimal (low-tier DB), and free tools (GitHub, GitHub actions) will be used. SMS free tier system can be used.

Challenges:

- Clinics may not afford high subscription fees.
- Students have limited budget for hosting and SMS free tier credits.

Solution:

- Adopt a freemium model: basic booking free, small fee per confirmed booking.
- For pilot/testing: use sandbox SMS and free-tier hosting.
- Clear ROI for clinics: fewer no-shows, more billable visits and less admin making for cost savings.

Legal Feasibility

Description:
The system must comply with South Africa's POPIA law (Protection of Personal Information Act). This includes consent, minimal data storage, secure retention, and right to access or delete data.

Challenges:

- Storing patient personal data safely.
- Managing patient consent and audit history.

Solution:

- Consent flags in database before reminders can be sent.
- Audit logs for all data access and changes.
- Use role-based access control to limit who can see patient data when needed.

Schedule Feasibility

Description:
The project is planned across 6 sprints (~6 weeks), with each member leading one sprint. Milestones are aligned with course deadlines.

Challenges:

- Risk of slippage if tasks are too complex for one sprint.
- Limited student time (parallel courses).
- Distance learning students might have jobs experiencing this now.

Solution:

- Use Agile with 1-week sprints and frequent demos.
- Fix scope per sprint and make tasks less if needed.
- Regular stand-ups and GitHub tracking for accountability.

2.4 Project planning

Scope (in):

- Booking: Create/modify/cancel, conflict detection, visit types.
- Reminders: SMS/email, templates, opt-out.
- Queue: Check-in, priority tags, live wait estimate, notifications.
- Triage: Short pre-booking questionnaire, urgent flagging.
- Records: Basic patient profile and visit notes (not full EMR).
- Analytics: No-show %, utilization, peak hours.
- Security: RBAC, audit logs, consent.

Scope (out for Phase 2):

- Full billing/ICD-10 coding, e-prescriptions, insurance claims, telehealth video, multi-clinic inventory.

Deliverables:

- Planning docs (this set), SRS, ERD/Schema, API spec, wireframes, working MVP, test evidence, demo.

Quality plan:

- Definition of done: Unit + integration tests passed, RBAC enforced, error logging, UX check, data validation.
- Reviews: End-of-sprint demo + code review.
- Risk register (top 5):
- Messaging API delays: Start with SMS only; abstract provider.
- Offline sync conflicts: Define “last-writer wins + manual resolution.”
- Data breach risk: Least-privilege, hashed secrets, TLS, minimal PII.
- Timeline slippage: Fix scope per sprint; timebox and descale if needed.
- Team bandwidth: Pairing, daily stand-ups, shared ownership.
- Communication plan:
- Cadence: Weekly sprint review; daily check-ins.
- Channels: GitHub issues/boards, Teams/WhatsApp, shared docs.

2.5 Project scheduling

Work breakdown structure (WBS)

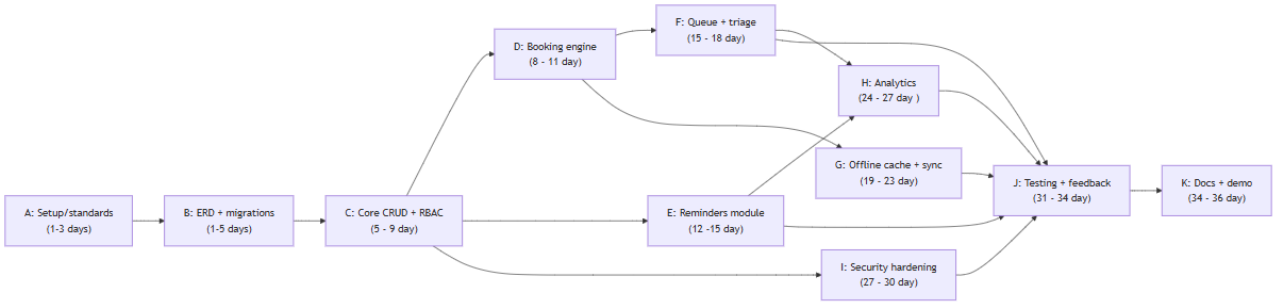
1. Inception & setup
- Repo, CI/CD, environments, coding standards
2. Data & core backend
- ERD, migrations, patient/doctor/appointment CRUD, RBAC, audit
3. Booking & calendar
- Create/modify/cancel, conflict engine, doctor availability
4. Reminders
- Templates, scheduler, delivery status, opt-out
5. Queue & check-in
- Triage priority, live queue, wait-time calc, notifications
6. Offline-first
- Local cache, sync queue, conflict resolution rules
7. Analytics
- No-show %, utilization, peak hours dashboards
8. Security & compliance
- Consent capture, data export/delete, logs
9. Testing & hardening
- Unit/integration/UI tests, performance, UAT
10. Docs & demo
- User guide, admin guide, deployment guide, demo script

PERT overview (key activities with optimistic/most likely/pessimistic days)

- A: Setup/standards:
- B: ERD + migrations: (depends A)
- C: Core CRUD + RBAC: (depends B)
- D: Booking engine: (depends C)
- E: Reminders module: (depends C)
- F: Queue + triage: (depends D)
- G: Offline cache + sync: (depends C, D)
- H: Analytics: (depends D, E, F)
- I: Security hardening: (parallel post-C)
- J: Testing + User feedback: (depends E, F, G, H, I)
- K: demos+ documentation: (final)

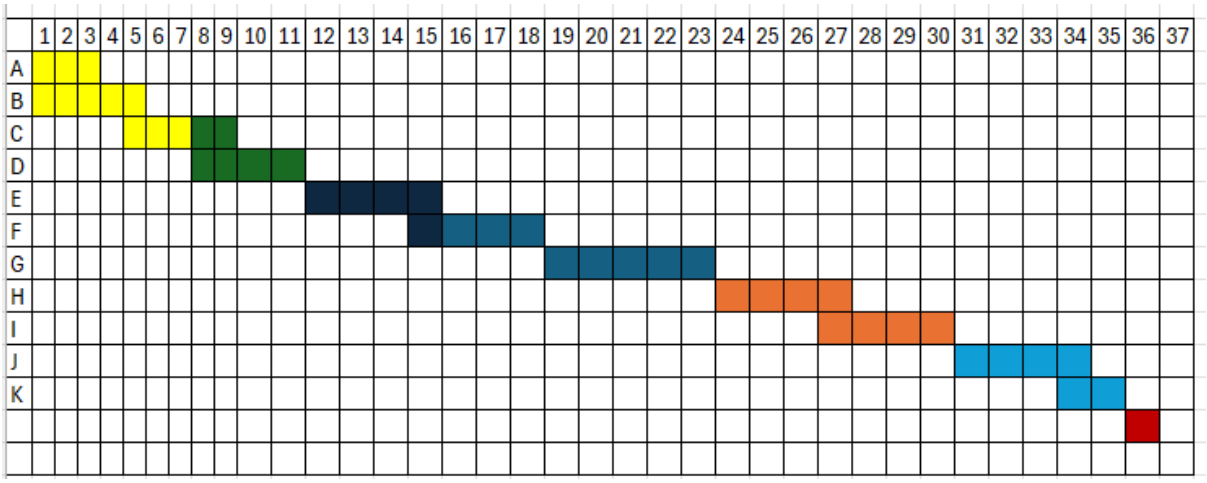
PERT Chart — Doctor Appointment System

Each node shows Task ID + expected duration .



Gantt (6 sprints split into 5–7 days each)

- Sprint 1 (Days 1 - 5): A, B, start C (Yellow)
- Sprint 2 (Days 6–11): Finish C, D (green)
- Sprint 3 (Days 12–16): E, start F (Navy)
- Sprint 4 (Days 17–23): Finish F, G (blue)
- Sprint 5 (Days 24–30): H, I (orange)
- Sprint 6 (Days 31–37): J, K, release candidate and demo (light blue and red)



Milestones:

- M1: DB + RBAC running
- M2: End-to-end booking flow
- M3: Reminders live
- M4: Queue + triage live
- M5: Analytics live
- M6: Final tested build + demo

2.6 Software Requirements Specification (SRS)

1. Introduction

- Purpose: Define functional and non-functional requirements for a low-cost, clinic-friendly appointment and office management system with multi-channel booking and offline tolerance.
- Scope: Single clinic initially; extensible to multi-doctor, multi-room; minimal patient record; no e-prescribing/billing in Phase 2.
- Definitions: No-show, triage, offline cache, RBAC, consent, PHI/PII.
- Stakeholders: Patients, receptionists, doctors, clinic manager, system admin.

2. Overall description

- User classes:
- Patient: Search doctor/time, book/cancel, receive reminders.
- Receptionist: Manage bookings, check-in, queue, patient profiles.
- Doctor: View schedule, mark complete/no-show, add brief visit notes.
- Manager: Reports/analytics, user management, settings.
- Admin: System configuration, roles, backups.
- Operating environment: Web app (modern browsers), Python backend (Fast API/Flask), SQLite/MySQL, SMS/email gateway.
- Design constraints: POPIA compliance, offline-first reception, lightweight UI.

References

Dennis, A. W. B. &. T. D., 2008. *Systems analysis and design: An object-oriented approach with UML*. s.l.:6th ed.

Fitzgerald, D. &., 2001. *Information systems development: methodologies*. s.l.:4th ed.

Garg, A. A. N. M. H. R.-A. M., 2020. *Effects of computerized clinical decision support systems on practitioner performance and patient outcomes: a systematic review*. s.l.:JAMA.

Laudon, K. &. L. J., 2020. *Management information systems: managing the digital firm*. s.l.:Pearson.

Republic of South Africa, 2013. *Protection of Personal Information Act 4 of 2013 (POPIA)*.. s.l.:Government Gazette.