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

SYSTEM DESIGN PHASE

4.1 Introduction

The design phase transforms the system requirements identified during the analysis phase into a structured blueprint that guides system implementation. This phase ensures that the Doctor Appointment & Office Management System is designed to be scalable, secure, and efficient for use by small and medium-sized clinics.The design emphasizes modularity, user-friendliness, and compliance with data protection standards such as POPIA. It also supports iterative Agile development, enabling continuous improvement and early feedback during implementation.

4.2 System Design (Description of Proposed System)

The Doctor Appointment & Office Management System is designed as a web-based platform that streamlines patient appointment booking, queue management, and doctor scheduling.The system aims to reduce administrative workload, minimize waiting times, and enhance patient satisfaction.

The system is composed of the following major modules:

1. Appointment Management – Allows patients and staff to schedule, modify, and cancel appointments.

2. Queue Management – Manages patient flow within the clinic and reduces waiting times.

3. Notification Service – Sends SMS or WhatsApp reminders and updates to patients.

4. Triage and Questionnaire Module – Enables patients to complete health forms before their appointment.

5. Analytics Dashboard – Provides visual insights into appointment trends and resource utilization.

6. Authentication and Role Management – Controls access based on user roles (Admin, Doctor, Receptionist, Patient).

7. Offline Sync Module – Ensures the system operates in offline mode and synchronizes data when reconnected.

Data Flow:

Patient information flows through the system from registration to appointment confirmation, queue management, and report generation and Data is securely transmitted via encrypted HTTPS protocols and stored in a central database.

4.3 Architectural Design (Software Architectural Design, Hardware Architectural Design, Network Architectural Design, class diagram)

4.3.1 Software Architectural Design

The system follows a three-tier architecture:

1. Presentation Layer (Frontend):
- Technologies: HTML, CSS, JavaScript
- Provides user interfaces for patients, doctors, and receptionists.

2. Application Layer (Backend):

-Technology: Python (Fast API)
-Handles business logic, appointment management, reminders, and validation.

3.Data Layer (Database):

- Technologies: SQLite (for local/offline support)
- Stores patient, doctor, appointment, and audit data.

4.3.2 Hardware Architectural Design

-Server: Hosts backend API and database.
-Minimum specs: Dual-core processor, 8GB RAM, 80GB SSD storage(more when needed).
-Client Devices: Desktop PCs, tablets, and smartphones.

4.3.3 Network Architectural Design

-Topology: Cloud-hosted or LAN-based architecture with secure HTTPS access.

Communication:

-Clients communicate with backend via REST APIs.

-Notifications sent through third-party SMS/Email gateway and WhatsApp when needed.

Connectivity:

-Supports offline queue mode for reception (syncs data when network returns).

4.3.4 Class Diagram

The class diagram defines the major entities and their relationships in the system.
Key classes include Patient, Doctor, Appointment, User, Notification, Queue Entry, and Triage Response.

Relationships:
- A Doctor can have many Appointments.
- A Patient can have multiple Appointments.
- Each Appointment generates one Queue Entry and may trigger multiple Notifications.
- The User class provides authentication and authorization for different roles.

4.4 Physical Design

4.4.1 Database Design

The physical database design defines the structure for data storage and retrieval.
The system uses MySQL for central data management and SQLite for offline operation.

Tables include:

- Patient (patient_id, name, contact, medical_history)
- Doctor (doctor_id, name, specialization, schedule)
- Appointment (appointment_id, patient_id, doctor_id, date, status)
- Notification (notification_id, appointment_id, message, timestamp)
- QueueEntry (queue_id, appointment_id, time_in, time_out)
- User (user_id, username, password_hash, role)

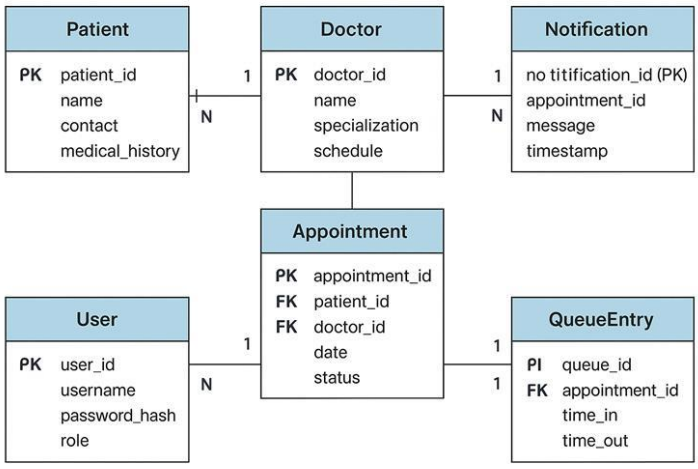
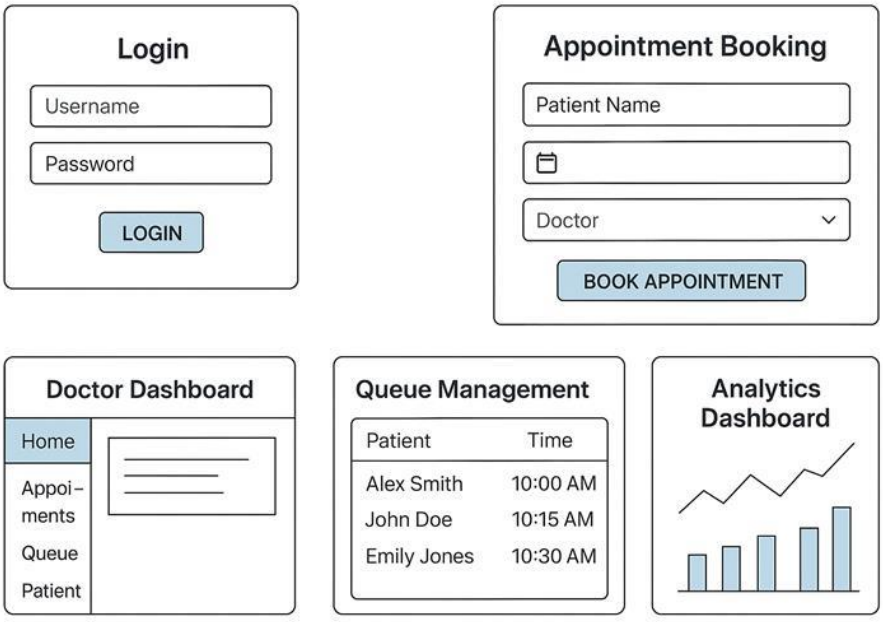


Figure 5 shows the Entity-Relationship Diagram (ERD) representing these tables and relationships.

4.4.2 User Interface Design

The system interface is designed for simplicity and ease of navigation.
It uses consistent color themes, clear icons, and responsive layouts to ensure usability across devices.

Key screens include:
- Login Page
- Appointment Booking Form
- Doctor Dashboard
- Queue Management Screen
- Analytics Dashboard



4.4.3 Security Design

Security is prioritized to ensure compliance with POPIA and protect patient data.
All user data is transmitted through HTTPS and stored using strong encryption mechanisms.
Password storage uses crypt hashing, and access control is enforced through role-based permissions.
Audit logs record all user activity for accountability.

4.4.4 System Integration

All modules interact through a RESTful API framework, ensuring seamless communication between components.
The integration layer handles request validation, authentication, and data transformation.
Figure 7 presents the sequence diagram showing the process from appointment booking to notification dispatch.

4.5 Database Design

The database design defines the structure of how information will be stored, related, and retrieved within the Doctor Appointment & Office Management System.

A relational database model is used to ensure data integrity, normalization, and scalability.

4.5.1 Logical Database Design

The system uses MySQL as the central database, with SQLite as a backup for offline functionality. The key entities and relationships are illustrated in the Entity Relationship Diagram (ERD) described below.

Main Entities and Attributes:

Entity	Attributes	Description
Patient	patient_id (PK), full_name, gender, date_of_birth, contact_number, email, address, medical_history	Stores patient demographic and medical data.
Doctor	doctor_id (PK), full_name, specialization, contact_number, email, availability	Contains information about doctors and their working schedules.
Appointment	appointment_id (PK), patient_id (FK), doctor_id (FK), appointment_date, time_slot, status, reason	Tracks appointments between patients and doctors.
User	user_id (PK), username, password_hash, role	Stores system login credentials and access roles.
Notification	notification_id (PK), appointment_id (FK), message, status, sent_at	Manages communication messages sent to patients.
Queue	queue_id (PK), appointment_id (FK), time_in, time_out, position	Tracks patients in the waiting queue.
Audit_Log	log_id (PK), user_id (FK), action, timestamp	Records all system activities for accountability.

4.5.2 Relationships

- One Doctor can have many Appointments.
- One Patient can have many Appointments.
- Each Appointment generates one Queue entry and multiple Notifications.
- Each User action is logged in the Audit_Log table.

4.5.3 Physical Database Design

- Primary Keys (PK): Uniquely identify records (e.g., patient_id, doctor_id).
- Foreign Keys (FK): Enforce referential integrity (e.g., appointment_id in Notification references Appointment table).
- Indexes: Applied on appointment_date and doctor_id for faster search performance.
- Data Types: Optimized for efficiency (e.g., VARCHAR(100) for text fields, DATETIME for timestamps).

4.6 Program Design (Program Pseudo code)

A. Login Module

```
DISPLAY LoginScreen
PROMPT username, password

IF validateCredentials(username, password) == TRUE THEN
  userRole = getUserRole(username)
  REDIRECT to Dashboard(userRole)
ELSE
  DISPLAY "Invalid username or password"
END IF
```

B. Patient Registration

```
DISPLAY PatientRegistrationForm
INPUT patientDetails

IF validateInput(patientDetails) == TRUE THEN
  patientID = generateUniqueID()
  SAVE patientRecord(patientID, patientDetails)
  DISPLAY "Registration successful"
ELSE
  DISPLAY "Error: Invalid input"
END IF
```

C. Appointment Scheduling

```
DISPLAY ListOfDoctors
SELECT doctor, timeSlot

IF isSlotAvailable(doctor, timeSlot) == TRUE THEN
  appointmentID = createAppointment(patientID, doctor, timeSlot)
  ADD toQueue(appointmentID)
  DISPLAY "Appointment Confirmed"
ELSE
  DISPLAY "Selected slot is unavailable"
END IF

// Reschedule or Cancel
IF userChoosesReschedule THEN
  SELECT newDoctor, newTime
  UPDATE appointment(appointmentID, newDoctor, newTime)
  UPDATE queue accordingly
ELSE IF userChoosesCancel THEN
  CANCEL appointment(appointmentID)
  REMOVE fromQueue(appointmentID)
END IF
```

D. Queue Management

```
FUNCTION manageQueue(appointmentID):
  ADD patient to queue based on appointment time or arrival
  UPDATE queue positions

FUNCTION callNextPatient():
  nextPatient = getNextInQueue()
  UPDATE status(nextPatient, "In Consultation")
  REMOVE nextPatient from queue
```

E. Notification Module

```
upcomingAppointments = getUpcomingAppointments()

FOR each appointment IN upcomingAppointments:
  message = generateReminder(appointment)
  SEND message via preferredChannel(appointment.patient)
  IF patientResponds THEN
    UPDATE appointmentStatus(appointmentID, response)
  END IF
END FOR
```

F. Reporting & Analytics

```
DISPLAY ReportOptions
SELECT reportType

data = retrieveReportData(reportType)
stats = calculateStatistics(data)  // e.g., no-shows, utilization, wait times

DISPLAY stats
IF userWantsExport THEN
  EXPORT report(stats)
END IF
```

G. Offline Mode & Sync

```
IF detectConnectivity() == FALSE THEN
  STORE dataLocally(actions)
  DISPLAY "Offline mode: changes stored locally"
ELSE
  SYNC localData with server
  RESOLVE conflicts
  LOG sync actions
END IF
```

4.7 Interface Design (Menu Interface Design, Input Design, Output Design)

A. Menu Interface Design

- **Patient Menu:** Book Appointment | View/Cancel Booking | Contact Clinic
- **Receptionist Menu:** Manage Appointments | Queue View | Add Patient | Reports
- **Doctor Menu:** My Schedule | Patient Notes | Mark Complete/No-show
- **Manager/Admin Menu:** User Management | System Settings | Reports | Backup

B. Input Design

- Mobile-friendly forms for booking, check-in, and patient registration.
- Validate all required fields: date/time, ID, phone number, email, consent.
- Provide dropdowns or selection lists for doctors, time slots, and appointment types.

C. Output Design

- Appointment confirmation screen after booking.
- Queue dashboard showing patient positions and waiting times.
- Analytics reports: no-shows, utilization, peak hours.
- Printable summaries for doctors and patients.

4.8 Security back up design (Software concern).

Security Measures:

- POPIA compliance through data minimization and consent tracking.
- TLS encryption for all communications.
- Store passwords as hashed value.
- Role-based access (Reception, Doctor, Manager, Admin).
- Audit logging of sensitive operations (edit/delete patient data).

Backup & Recovery:

- Daily automated database backup to encrypted cloud storage.
- Manual export available for managers (CSV or SQL dump).
- Restore functionality for recovery after data loss.
- Versioning to ensure recovery from the most recent stable copy.

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.