**UNIVERSITY OF MINDANAO**
*College of Computing EducationDepartment of Information Technology*

# Python-Based Hospital Scheduling and Billing Management System

A Project Document Presented to:
**Modesto C. Tarrazona**
IT5/L Instructor

**In Partial Fulfillment of the Requirements for the Course:**
**IT5 – IT Elective 2**

Prepared by:
Kate Kirslet V. Sabio
Subject Code:
2868

**Date Submitted:**
December 15, 2025

# Table of Contents

**Page Title** **Page Number**

# I. Introduction

## 1.1 Brief Overview of the Project

MediSked is a comprehensive desktop-based hospital scheduling and billing management system designed to streamline administrative workflows in small to medium-sized healthcare facilities. The application provides an integrated platform for managing patient appointments, doctor availability, medical records, and billing transactions through an intuitive graphical interface.

## 1.2 Background of the Study

Healthcare facilities, particularly small clinics and hospitals, continue to rely heavily on manual, paper-based administrative systems for scheduling appointments and managing billing processes. This traditional approach creates operational inefficiencies, increases the likelihood of human error, and limits staff productivity. The need for a digital solution that consolidates patient information, appointment scheduling, and financial transactions into a single accessible system has become increasingly critical as healthcare organizations seek to improve service delivery while maintaining data security and operational transparency.

## 1.3 Importance of the Project

The MediSked system addresses fundamental operational challenges faced by healthcare facilities:

- **Enhanced Efficiency**: Automates repetitive administrative tasks, allowing healthcare staff to focus on patient care rather than paperwork
- **Error Reduction**: Minimizes scheduling conflicts, double bookings, and billing discrepancies through systematic validation and real-time data synchronization
- **Improved Patient Experience**: Reduces wait times and improves service quality through streamlined appointment management
- **Financial Accuracy**: Ensures accurate tracking of payments and outstanding balances, reducing revenue leakage
- **Data Security**: Implements role-based access control to protect sensitive patient information and maintain DPA-compliant data handling practices
- **Accountability**: Provides comprehensive activity logging to track all system operations and user actions for audit purposes

### 1.4 Tools and Technologies

**Programming Language**

Python 3.x - Core development language providing robust libraries and cross-platform compatibility.

**Database Management**

SQLite - Lightweight, serverless relational database for storing patient records, appointments, user accounts, and billing information.

**User Interface Framework**

CustomTkinter - Modern, customizable GUI framework built on top of Tkinter, providing enhanced visual aesthetics and responsive design elements.

**Development Environment**

Visual Studio Code - Integrated Development Environments (IDEs) for code development and debugging.

**Additional Libraries:**

- datetime - Date and time manipulation for appointment scheduling
- hashlib - Secure password hashing for user authentication
- os - Operating system interface for file and directory operations
- csv - Export functionality for generating reports
- bmp -Export functionality for generating receipts

# II. Statement of the Problem

Healthcare facilities operating with manual administrative systems face several critical operational challenges:

### 2.1 Inefficient Manual Processes

Healthcare staff expend excessive time on administrative tasks such as manually scheduling appointments, searching through physical records, and processing payments. This reduces the time available for actual patient care and creates bottlenecks during peak hours when multiple patients require simultaneous attention. The cumulative effect of these inefficiencies results in longer patient wait times, staff burnout, and reduced overall facility throughput.

### 2.2 Scheduling Conflicts and Double Bookings

Without a centralized system to track doctor availability in real-time, receptionists frequently encounter scheduling conflicts, resulting in double bookings, patient dissatisfaction, and wasted doctor time. The lack of visibility into each doctor's schedule across different staff members compounds this issue, as there is no single source of truth for appointment availability. This fragmentation leads to miscommunication between staff members and creates confusion when patients arrive for appointments that were inadvertently scheduled simultaneously.

### 2.3 Disorganized Patient Information

Patient records scattered across paper files make it difficult to quickly retrieve medical histories, previous appointments, or billing information. This fragmentation leads to longer wait times, potential medical errors due to incomplete information, and poor continuity of care. Healthcare providers cannot efficiently access a patient's complete history, which compromises decision-making and treatment planning.

### 2.4 Billing and Payment Tracking Challenges

Manual billing processes are prone to errors, lost invoices, and delayed payments. Cashiers struggle to maintain accurate records of outstanding balances, completed payments, and financial reconciliation, resulting in revenue leakage and accounting discrepancies. The absence of automated payment tracking makes it difficult to identify unpaid appointments, follow up with patients, and generate comprehensive financial reports for management review.

### 2.5 Lack of Access Control and Accountability

Paper-based systems offer no mechanism to restrict access to sensitive information or track who performed specific actions. This creates security vulnerabilities and makes it impossible to audit staff activities or maintain proper data confidentiality in compliance with healthcare privacy standards. Without proper access controls, there is increased risk of unauthorized access to patient information, potential DPA violations, and difficulty in identifying the source of errors or discrepancies.

## III. Objectives of the Study

### 3.1 General Objective

To design and implement a secure, role-based desktop application that automates hospital scheduling and billing operations, thereby improving operational efficiency, data accuracy, and patient service quality in healthcare facilities.

### 3.2 Specific Objectives

1. **To develop a role-based desktop application using Python**, CustomTkinter, and SQLite that supports distinct workflows for Admin, Doctor, Cashier, and Receptionist roles, ensuring each user has access only to functions relevant to their responsibilities.
2. **To centralize appointment data, doctor availability, and payment status** in a single SQLite database, eliminating data redundancy and ensuring all staff members work from a consistent, real-time view of hospital operations.

3. **To minimize scheduling and billing errors** by implementing automated validation rules that prevent double-booking of appointments, enforce proper data entry, and clearly distinguish between paid and unpaid transactions.
4. **To enhance security and accountability** through admin-controlled user account creation, secure password hashing, and comprehensive activity logging that tracks all user actions for audit and compliance purposes.
5. **To process and display appointment information** using Python's datetime library and CustomTkinter widgets, presenting schedules in an intuitive calendar format that allows receptionists to quickly identify available time slots.
6. **To generate financial reports and billing summaries** using Python's data processing capabilities, enabling cashiers and administrators to track revenue, identify outstanding payments, and export financial data to CSV format for external analysis.
7. **To improve user experience** through a modern, responsive graphical interface built with CustomTkinter that reduces training time, minimizes user errors, and provides clear visual feedback for all operations.

## IV. Scope and Limitations

### 4.1 Scope

The MediSked system encompasses the following features and functionalities:

**User Management:**

- Admin-controlled creation of Doctor, Receptionist, and Cashier accounts
- Secure login authentication with role-based access control
- Pre-configured admin account for initial system setup

**Appointment Management:**

- Calendar-based appointment scheduling interface
- Real-time doctor availability tracking
- Prevention of double-booking conflicts
- Appointment rescheduling and cancellation
- Support for multiple appointment types and consultation services

**Doctor Management:**

- Doctor availability calendar configuration
- Time slot creation and management
- Personal appointment dashboard with upcoming schedule
- Completed appointment records and patient history

**Patient Records:**

- Patient information storage (name, contact number, address)
- Appointment history tracking

- Integration with billing information

**Billing and Payment Processing:**

- Point-of-sale (POS) transaction interface for cashiers
- Payment verification using appointment barcodes
- Automatic calculation of charges and change
- Clear distinction between paid and unpaid appointments
- Payment history and receipt generation

**Reporting and Analytics:**

- Admin dashboard with system-wide statistics
- Doctor-specific appointment analytics
- Financial reports (daily earnings, monthly revenue)
- CSV export functionality for external data analysis

**Activity Logging:**

- Comprehensive logging of all user actions
- Login/logout tracking
- Appointment creation, modification, and deletion logs
- Payment confirmation logs
- System event tracking for audit purposes

## 4.2 Limitations

**Deployment:**

- Desktop-only application; no web or mobile interface
- Single-user concurrent access (no multi-user network support)
- Requires local installation on each workstation

**Integration:**

- No integration with external laboratory or pharmacy systems
- No electronic medical records (EMR) functionality beyond basic patient information
- No integration with insurance claim processing systems

**Communication:**

- No automated patient notification system (SMS/email reminders)
- No online appointment booking portal for patients

**Technical:**

- SQLite database limitations for extremely high transaction volumes
- No built-in backup automation (manual backup procedures required)
- Limited support for handling concurrent database operations

**Clinical Features:**

- No electronic prescription generation
- No medical imaging or diagnostic report management
- No clinical decision support or drug interaction checking

**Scalability:**

- Designed for small to medium-sized clinics (not enterprise hospitals)
- Limited support for multiple facility locations or departments.

## V. Data Flow Diagrams (DFD)

### 5.1 DFD Level 0 (Context Diagram)

The context diagram represents the MediSked system as a single process, showing the primary external entities and their interactions:

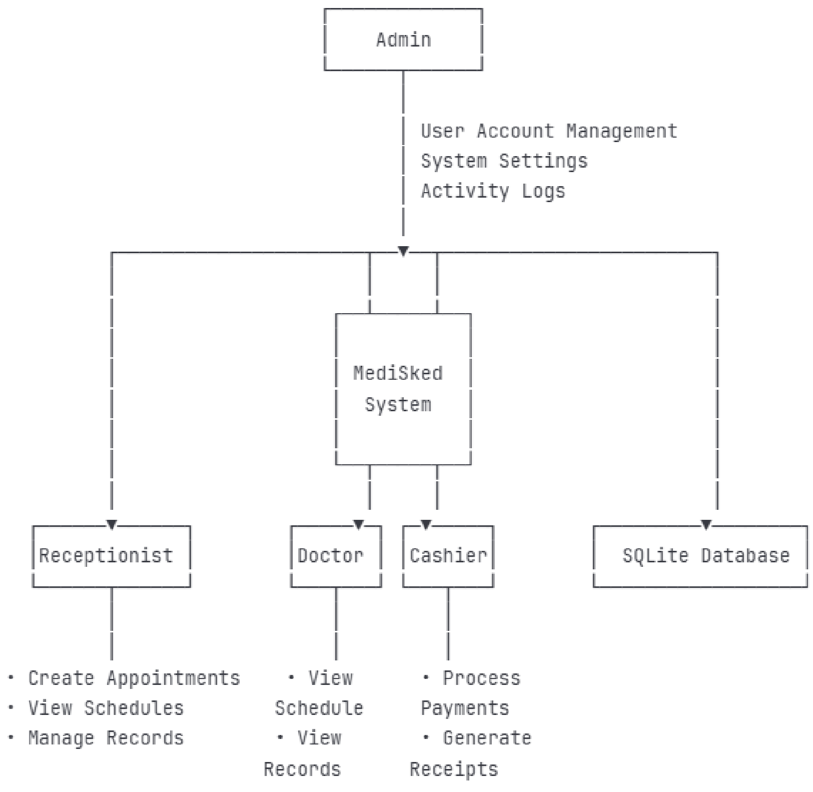


*Figure 1. DFD Level 0*

**External Entities:**

- **Admin**: Manages user accounts, system configuration, and monitors activity logs

- **Receptionist**: Creates appointments, manages patient information, and views schedules
- **Doctor**: Views personal schedule, appointment details, and patient records
- **Cashier**: Processes payments, verifies appointments, and generates billing reports
- **SQLite Database**: Stores all system data including users, appointments, patients, and transactions

**5.2 DFD Level 1 (Process Decomposition)**

The Level 1 diagram breaks down the MediSked system into major processes:

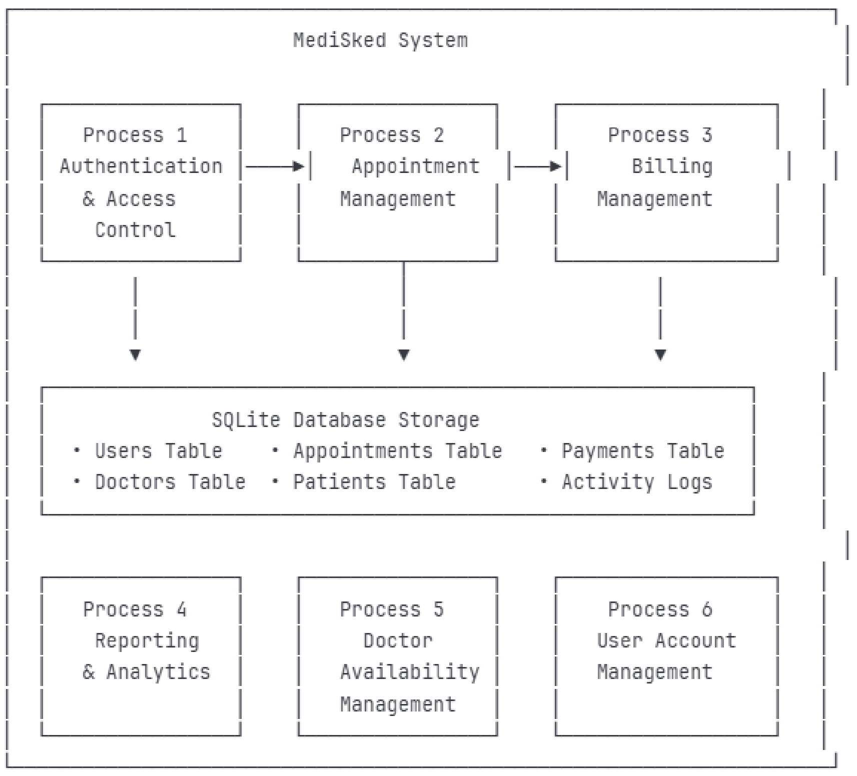

MediSked System
Process 1 Authentication & Access Control
Process 2 Appointment Management
Process 3 Billing Management
SQLite Database Storage
• Users Table
• Appointments Table
• Payments Table
• Doctors Table
• Patients Table
• Activity Logs
Process 4 Reporting & Analytics
Process 5 Doctor Availability Management
Process 6 User Account Management

*Figure 2. DFD Level 1*

**Process Descriptions:**

**Process 1: Authentication & Access Control**

- Input: Username, password, role selection
- Process: Validates credentials, checks role permissions, creates user session
- Output: Access granted/denied, user dashboard redirect
- Data Store: Users table

**Process 2: Appointment Management**

- Input: Patient details, doctor selection, date/time, service type

- Process: Checks doctor availability, validates time slots, creates appointment record, generates barcode
- Output: Appointment confirmation, barcode, updated schedule
- Data Store: Appointments table, Patients table, Doctor_Availability table

**Process 3: Billing Management**

- Input: Appointment barcode, payment amount
- Process: Retrieves appointment details, calculates total and change, updates payment status
- Output: Payment receipt, updated appointment status
- Data Store: Payments table, Appointments table

**Process 4: Reporting & Analytics**

- Input: Date range, doctor filter, report type
- Process: Aggregates data, calculates statistics, formats output
- Output: Dashboard statistics, CSV reports, financial summaries
- Data Store: All tables

**Process 5: Doctor Availability Management**

- Input: Doctor selection, date, time slots
- Process: Creates/updates availability records, prevents conflicts
- Output: Updated availability calendar
- Data Store: Doctor_Availability table

**Process 6: User Account Management**

- Input: Username, password, role (Admin only)
- Process: Validates input, hashes password, creates user record
- Output: New user account confirmation
- Data Store: Users table

## VI. System Implementation

### 6.1 System Requirements

**Hardware Requirements:**

Minimum Configuration:

- Processor: Intel Core i3 or equivalent (2.0 GHz or higher)
- RAM: 4 GB
- Storage: 500 MB available disk space
- Display: 1366 x 768 resolution
- Input Devices: Keyboard and mouse

Recommended Configuration:

- Processor: Intel Core i5 or higher (2.5 GHz or higher)
- RAM: 8 GB or more
- Storage: 1 GB available disk space (SSD preferred)
- Display: 1920 x 1080 resolution or higher
- Input Devices: Keyboard, mouse, barcode scanner (optional for cashier stations)

**Software Requirements:**

**Operating System:**
- Windows 10/11 (64-bit)
- macOS 10.14 (Mojave) or later
- Linux (Ubuntu 20.04 LTS or equivalent)

**Runtime Environment:**
- Python 3.8 or higher

**Database:**
- SQLite 3 (included with Python)

### 6.2 Programming Tools

**Python Version:**
- Python 3.8+ (recommended: Python 3.10 or 3.11)

**Integrated Development Environment (IDE):**
- PyCharm Community/Professional Edition
- Visual Studio Code with Python extension

**Required Python Libraries:**

GUI Framework**:**
- customtkinter (v5.0.0+): Modern, customizable GUI framework

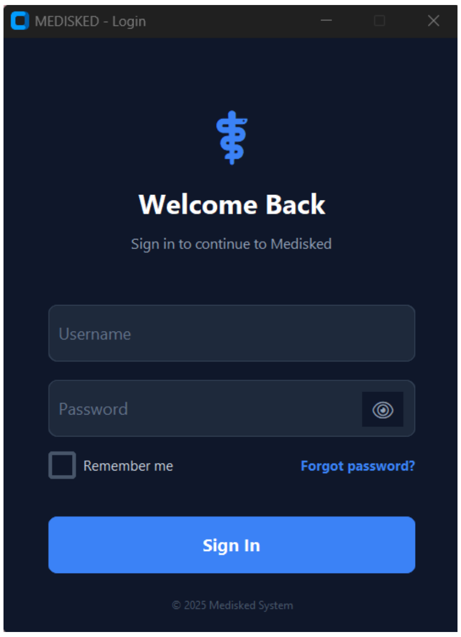
Database:
- sqlite3: Built-in Python module (no installation required)

**Standard Libraries:**

- **datetime**: Date and time handling for appointment scheduling
- **hashlib**: Secure password hashing (SHA-256)
- **os**: File system operations and path management
- **csv**: Report export functionality
- **bmp**: Receipt export functionality
- **random**: Barcode generation
- **tkinter**: Base GUI framework (bundled with Python)

**Optional Libraries for Enhancement**:
- **Pillow (PIL)**: Image processing for logos and icons
- **python-barcode**: Advanced barcode generation
- **pandas**: Data analysis and reporting

## VII. Screenshots


MEDISKED - Login

Welcome Back

Sign in to continue to Medisked

Username

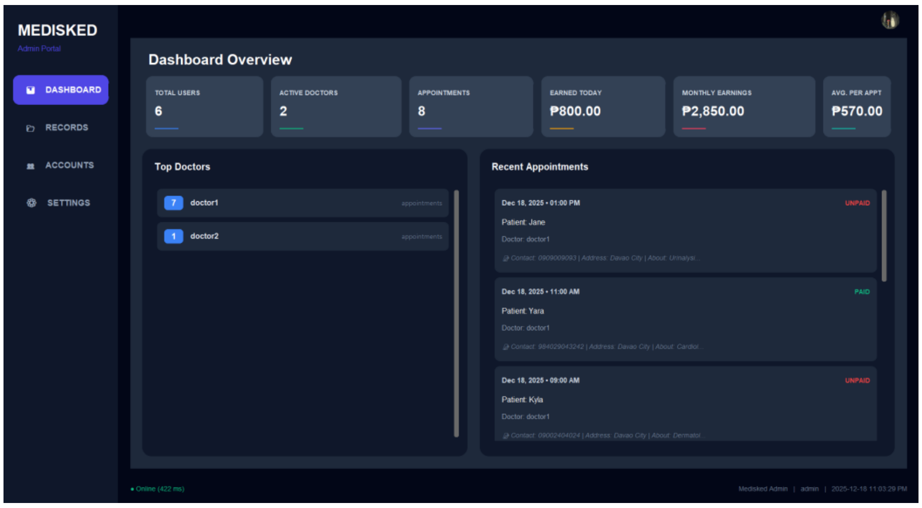
Password

Remember me

Forgot password?

Sign In

© 2025 Medisked System


MEDISKED
Admin Portal

DASHBOARD
RECORDS
ACCOUNTS
SETTINGS

Dashboard Overview

| TOTAL USERS | ACTIVE DOCTORS | APPOINTMENTS | EARNED TODAY | MONTHLY EARNINGS | AVG. PER APPT |
|---|---|---|---|---|---|
| 6 | 2 | 8 | ₱800.00 | ₱2,850.00 | ₱570.00 |

Top Doctors

| | | |
|---|---|---|
| 7 | doctor1 | appointments |
| 1 | doctor2 | appointments |

Recent Appointments

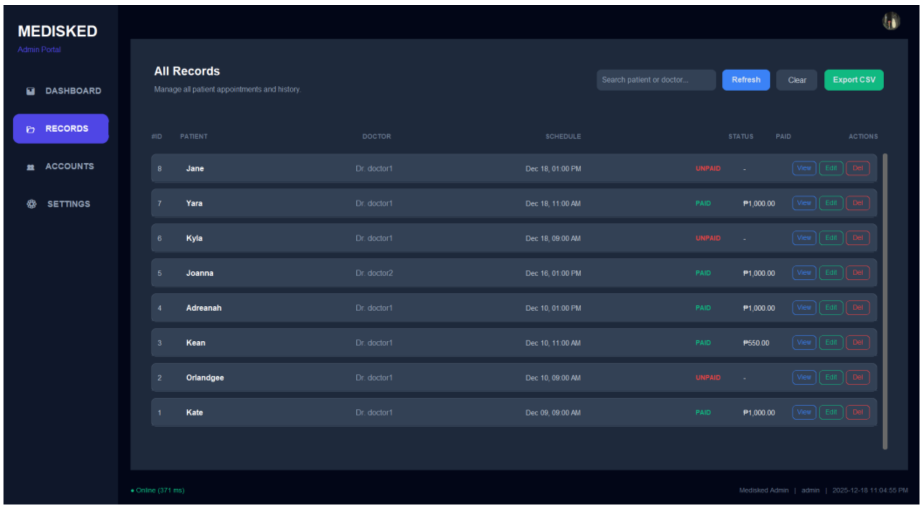
Dec 18, 2025 • 01:00 PM UNPAID
Patient: Jane
Doctor: doctor1
Contact: 0909009093 | Address: Davao City | About: Urinalysi...

Dec 18, 2025 • 11:00 AM PAID
Patient: Yara
Doctor: doctor1
Contact: 984029043242 | Address: Davao City | About: Cardiol...

Dec 18, 2025 • 09:00 AM UNPAID
Patient: Kyla
Doctor: doctor1
Contact: 09002404024 | Address: Davao City | About: Dermatol...

● Online (422 ms)

Medisked Admin | admin | 2025-12-18 11:03:29 PM


MEDISKED
Admin Portal

DASHBOARD
RECORDS
ACCOUNTS
SETTINGS

All Records

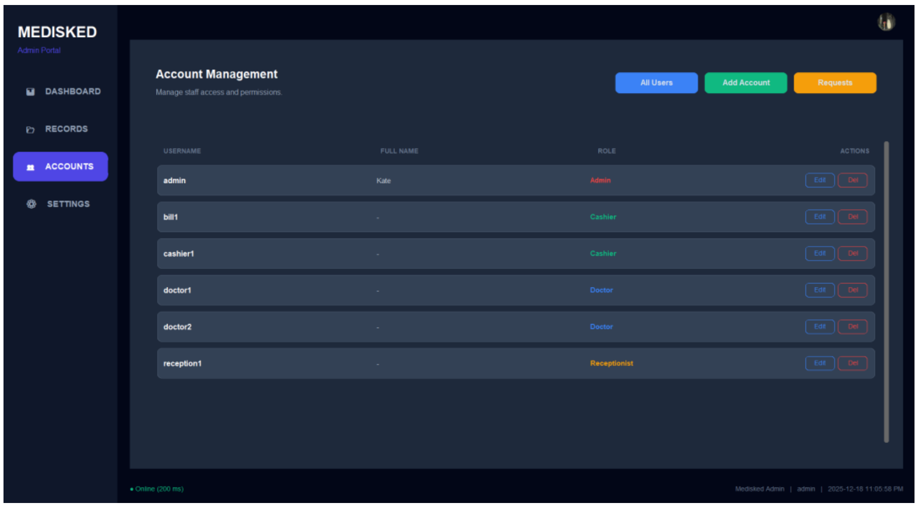
Manage all patient appointments and history.

Search patient or doctor... Refresh Clear Export CSV

| #ID | PATIENT | DOCTOR | SCHEDULE | STATUS | PAID | ACTIONS |
|---|---|---|---|---|---|---|
| 8 | Jane | Dr. doctor1 | Dec 18, 01:00 PM | UNPAID | - | View Edit Del |
| 7 | Yara | Dr. doctor1 | Dec 18, 11:00 AM | PAID | ₱1,000.00 | View Edit Del |
| 6 | Kyla | Dr. doctor1 | Dec 18, 09:00 AM | UNPAID | - | View Edit Del |
| 5 | Joanna | Dr. doctor2 | Dec 16, 01:00 PM | PAID | ₱1,000.00 | View Edit Del |
| 4 | Adreanah | Dr. doctor1 | Dec 10, 01:00 PM | PAID | ₱1,000.00 | View Edit Del |
| 3 | Kean | Dr. doctor1 | Dec 10, 11:00 AM | PAID | ₱550.00 | View Edit Del |
| 2 | Orlandgee | Dr. doctor1 | Dec 10, 09:00 AM | UNPAID | - | View Edit Del |
| 1 | Kate | Dr. doctor1 | Dec 09, 09:00 AM | PAID | ₱1,000.00 | View Edit Del |

● Online (371 ms)

Medisked Admin | admin | 2025-12-18 11:04:55 PM



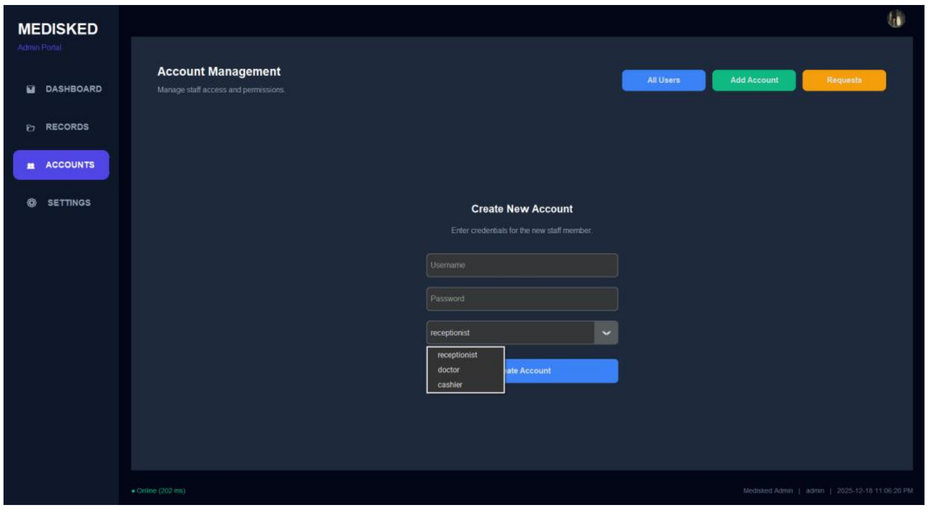
Account Management

Manage staff access and permissions.

| USERNAME | FULL NAME | ROLE | ACTIONS |
|---|---|---|---|
| admin | Kate | Admin | Edit Del |
| bill1 | - | Cashier | Edit Del |
| cashier1 | - | Cashier | Edit Del |
| doctor1 | - | Doctor | Edit Del |
| doctor2 | - | Doctor | Edit Del |
| reception1 | - | Receptionist | Edit Del |

● Online (200 ms)

Medisked Admin | admin | 2025-12-18 11:05:58 PM



Account Management

Manage staff access and permissions.

Create New Account

Enter credentials for the new staff member.

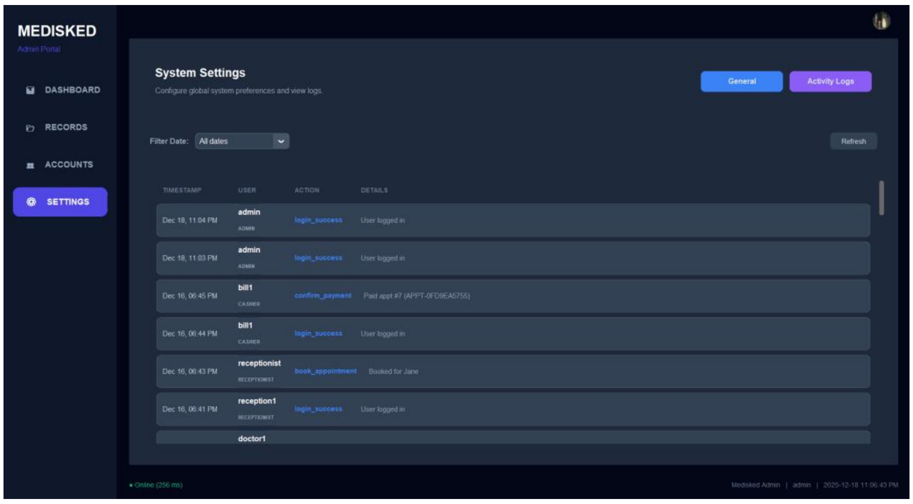
Username

Password

receptionist

● Online (202 ms)

Medisked Admin | admin | 2025-12-18 11:06:20 PM



System Settings

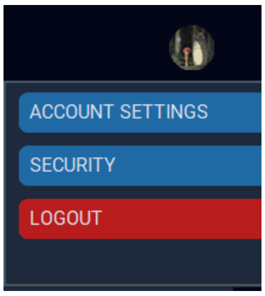
Configure global system preferences and view logs.

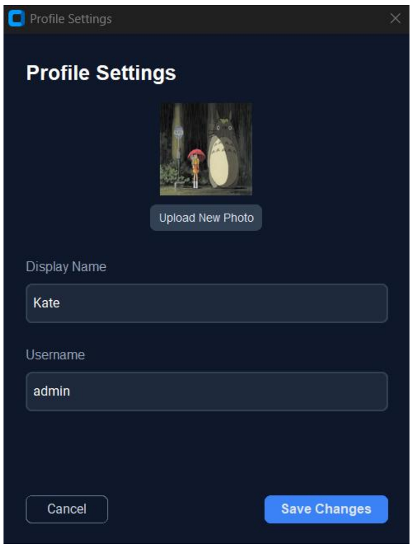
Filter Date: All dates

| TIMESTAMP | USER | ACTION | DETAILS |
|---|---|---|---|
| Dec 18, 11:04 PM | admin ADMIN | login_success | User logged in |
| Dec 18, 11:03 PM | admin ADMIN | login_success | User logged in |
| Dec 16, 06:45 PM | bill1 CASHIER | confirm_payment | Paid appt #7 (APPT-0FD9EA5755) |
| Dec 16, 06:44 PM | bill1 CASHIER | login_success | User logged in |
| Dec 16, 06:43 PM | receptionist RECEPTIONIST | book_appointment | Booked for Jane |
| Dec 16, 06:41 PM | reception1 RECEPTIONIST | login_success | User logged in |
| | doctor1 | | |

● Online (256 ms)

Medisked Admin | admin | 2025-12-18 11:06:43 PM

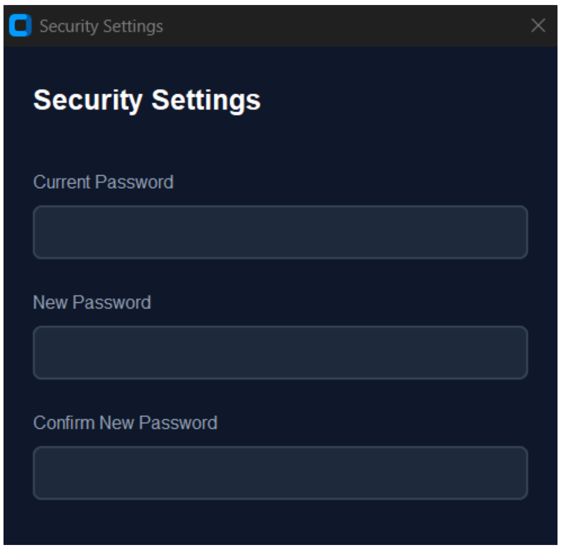




Profile Settings

Display Name: Kate

Username: admin

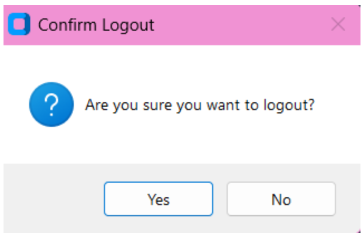


Security Settings

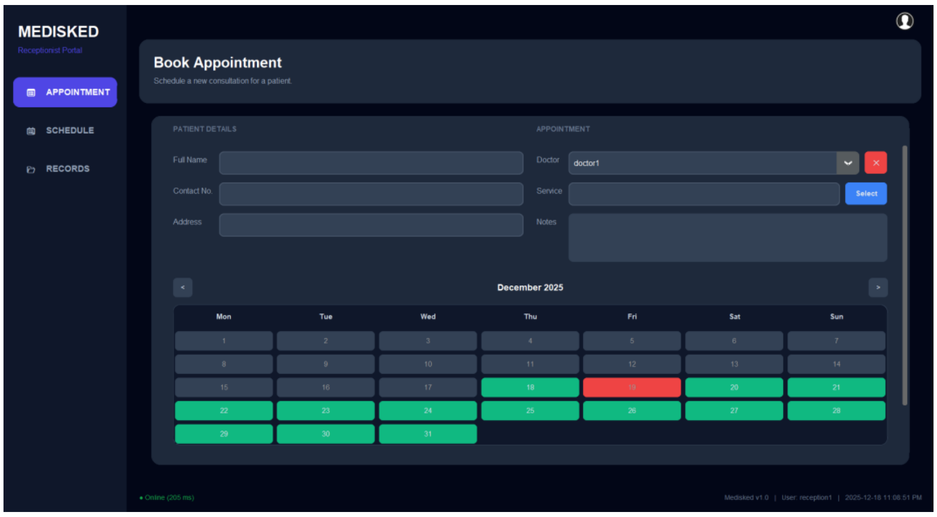
Current Password

New Password

Confirm New Password



Are you sure you want to logout?



Book Appointment

Schedule a new consultation for a patient.

PATIENT DETAILS

Full Name

Contact No.

Address

APPOINTMENT

Doctor: doctor1

Service

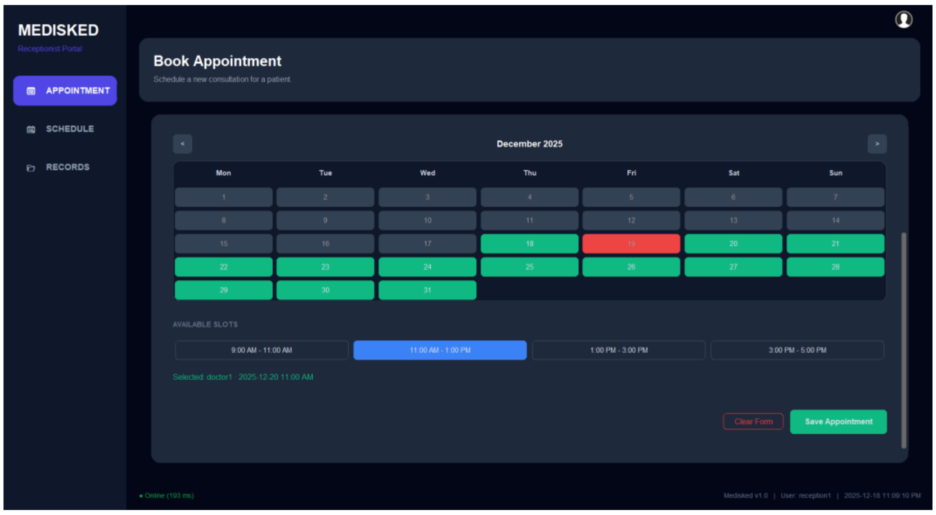
Notes

December 2025

| Mon | Tue | Wed | Thu | Fri | Sat | Sun |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | | | | |

● Online (205 ms)

Medisked v1.0 | User: reception1 | 2025-12-18 11:08:51 PM



Book Appointment

Schedule a new consultation for a patient

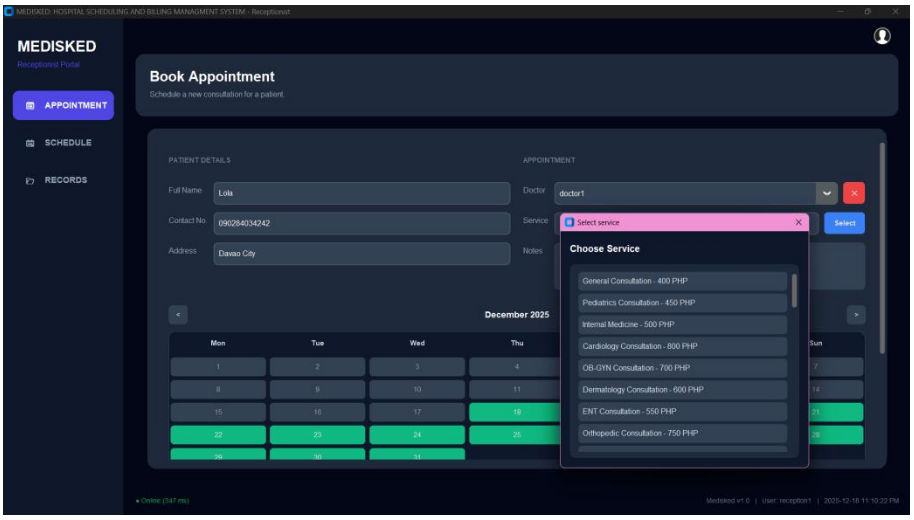
| Mon | Tue | Wed | Thu | Fri | Sat | Sun |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | | | | |

AVAILABLE SLOTS

9:00 AM - 11:00 AM | 11:00 AM - 1:00 PM | 1:00 PM - 3:00 PM | 3:00 PM - 5:00 PM

Selected: doctor1 · 2025-12-20 11:00 AM

● Online (193 ms)    Medisked v1.0 | User: reception1 | 2025-12-18 11:09:10 PM


MEDISKED: HOSPITAL SCHEDULING AND BILLING MANAGMENT SYSTEM - Receptionist

Book Appointment

Schedule a new consultation for a patient

PATIENT DETAILS

Full Name: Lola
Contact No: 090284034242
Address: Davao City

APPOINTMENT

Doctor: doctor1
Service
Notes

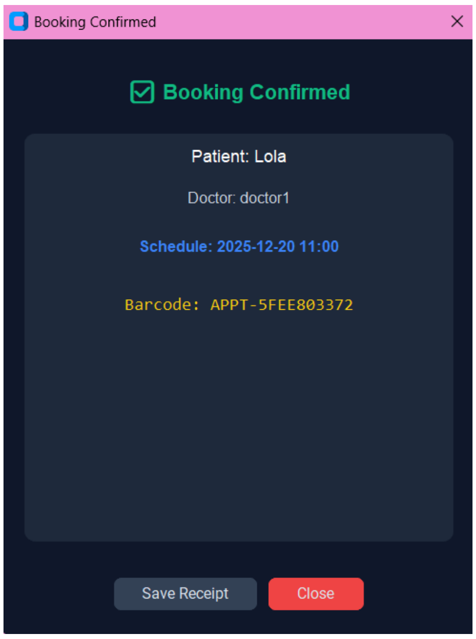
Select service

Choose Service

- General Consultation - 400 PHP
- Pediatrics Consultation - 450 PHP
- Internal Medicine - 500 PHP
- Cardiology Consultation - 800 PHP
- OB-GYN Consultation - 700 PHP
- Dermatology Consultation - 600 PHP
- ENT Consultation - 550 PHP
- Orthopedic Consultation - 750 PHP

| Mon | Tue | Wed | Thu | Sun |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 7 |
| 8 | 9 | 10 | 11 | 14 |
| 15 | 16 | 17 | 18 | 21 |
| 22 | 23 | 24 | 25 | 28 |
| 29 | 30 | 31 | | |

● Online (347 ms)    Medisked v1.0 | User: reception1 | 2025-12-18 11:10:22 PM


Booking Confirmed

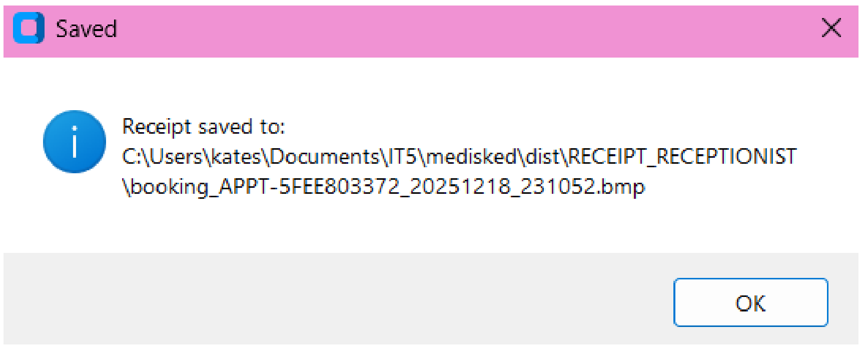
Booking Confirmed

Patient: Lola

Doctor: doctor1

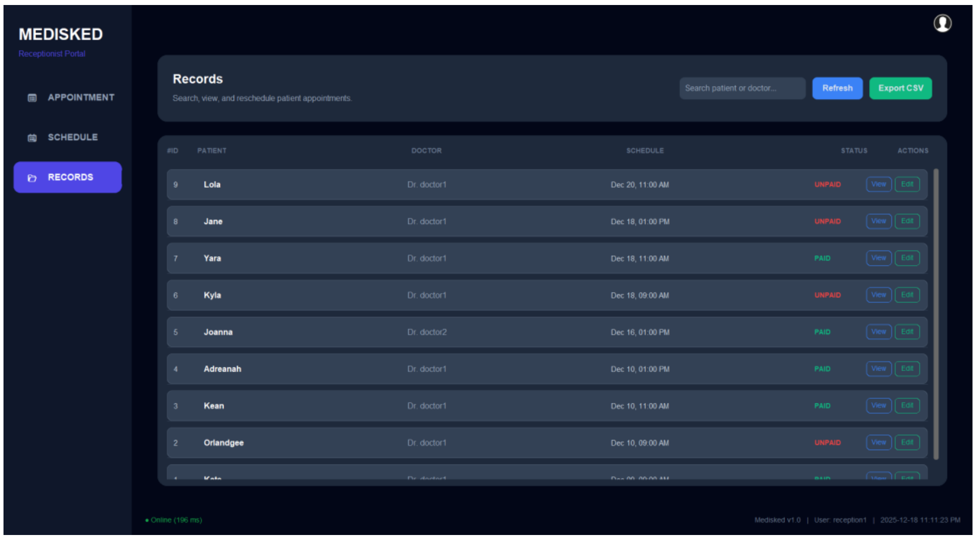
Schedule: 2025-12-20 11:00

Barcode: APPT-5FEE803372



Saved

Receipt saved to:
C:\Users\kates\Documents\IT5\medisked\dist\RECEIPT_RECEPTIONIST
\booking_APPT-5FEE803372_20251218_231052.bmp

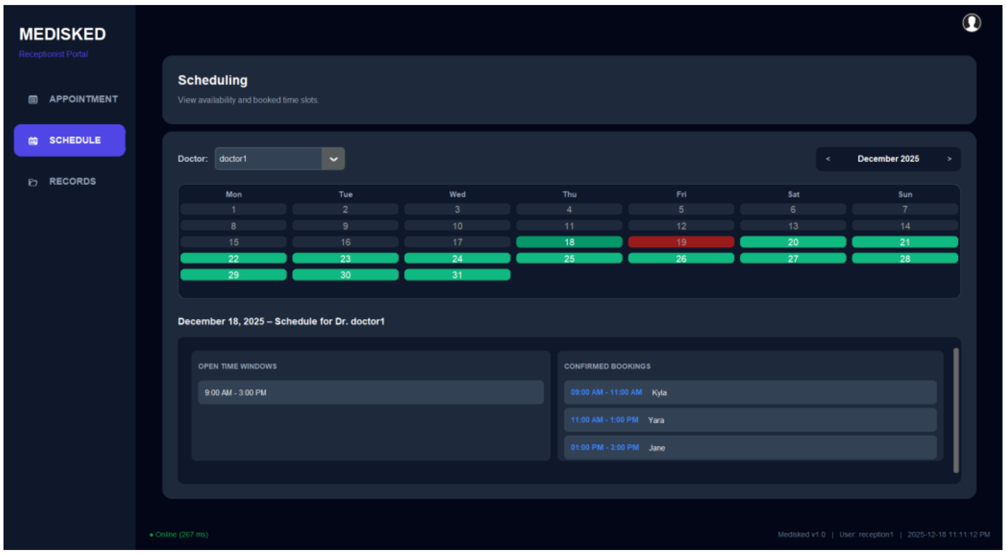


Records

Search, view, and reschedule patient appointments.

| #ID | PATIENT | DOCTOR | SCHEDULE | STATUS | ACTIONS |
|---|---|---|---|---|---|
| 9 | Lola | Dr. doctor1 | Dec 20, 11:00 AM | UNPAID | View Edit |
| 8 | Jane | Dr. doctor1 | Dec 18, 01:00 PM | UNPAID | View Edit |
| 7 | Yara | Dr. doctor1 | Dec 18, 11:00 AM | PAID | View Edit |
| 6 | Kyla | Dr. doctor1 | Dec 18, 09:00 AM | UNPAID | View Edit |
| 5 | Joanna | Dr. doctor2 | Dec 16, 01:00 PM | PAID | View Edit |
| 4 | Adreanah | Dr. doctor1 | Dec 10, 01:00 PM | PAID | View Edit |
| 3 | Kean | Dr. doctor1 | Dec 10, 11:00 AM | PAID | View Edit |
| 2 | Orlandgee | Dr. doctor1 | Dec 10, 09:00 AM | UNPAID | View Edit |

● Online (196 ms)    Medisked v1.0 | User: reception1 | 2025-12-18 11:11:23 PM



Scheduling

View availability and booked time slots.

Doctor: doctor1    < December 2025 >

| Mon | Tue | Wed | Thu | Fri | Sat | Sun |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | | | | |

December 18, 2025 – Schedule for Dr. doctor1

OPEN TIME WINDOWS

9:00 AM - 3:00 PM

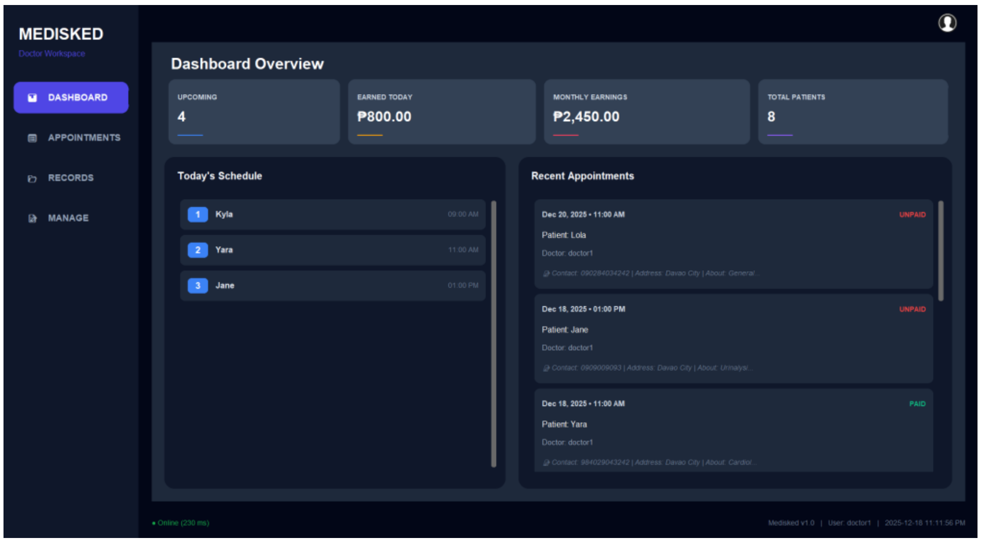
CONFIRMED BOOKINGS

09:00 AM - 11:00 AM Kyla

11:00 AM - 1:00 PM Yara

01:00 PM - 3:00 PM Jane

● Online (267 ms)    Medisked v1.0 | User: reception1 | 2025-12-18 11:11:12 PM



Dashboard Overview

UPCOMING
4

EARNED TODAY
₱800.00

MONTHLY EARNINGS
₱2,450.00

TOTAL PATIENTS
8

Today's Schedule

|  |  |  |
|---|---|---|
| 1 | Kyla | 09:00 AM |
| 2 | Yara | 11:00 AM |
| 3 | Jane | 01:00 PM |

Recent Appointments

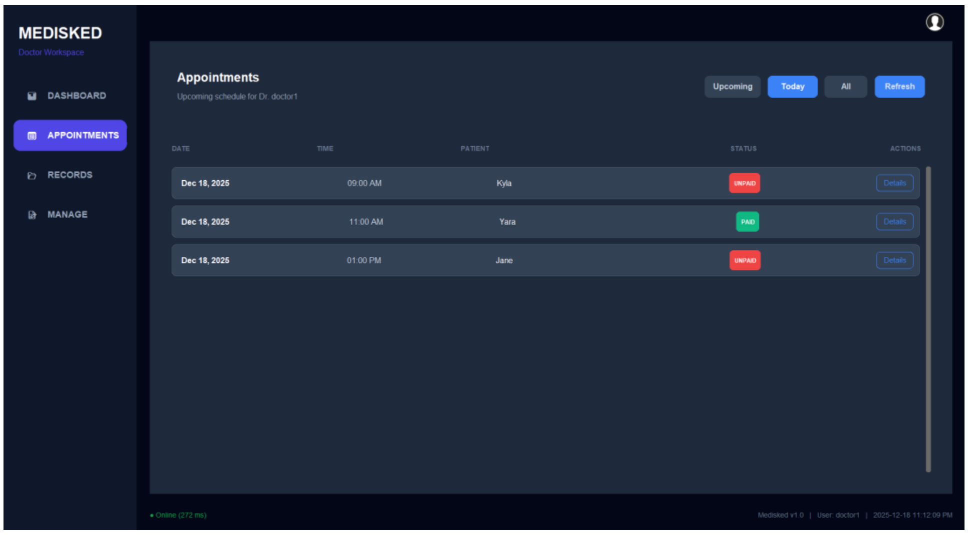
Dec 20, 2025 • 11:00 AM UNPAID
Patient: Lola
Doctor: doctor1
Contact: 090284034242 | Address: Davao City | About: General...

Dec 18, 2025 • 01:00 PM UNPAID
Patient: Jane
Doctor: doctor1
Contact: 0909009093 | Address: Davao City | About: Urinalysi...

Dec 18, 2025 • 11:00 AM PAID
Patient: Yara
Doctor: doctor1
Contact: 984029043242 | Address: Davao City | About: Cardiol...

• Online (230 ms)

Medisked v1.0 | User: doctor1 | 2025-12-18 11:11:56 PM



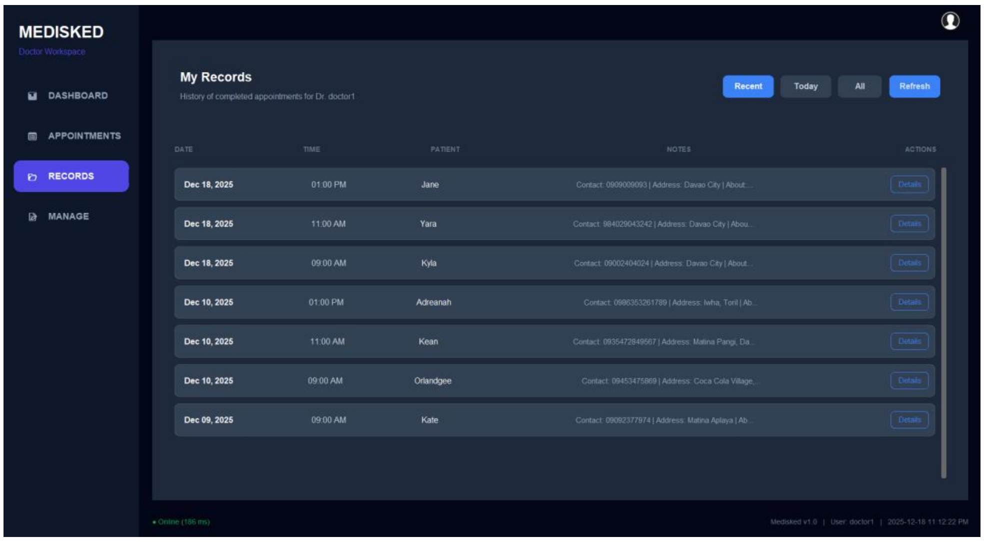
Appointments

Upcoming schedule for Dr. doctor1

| DATE | TIME | PATIENT | STATUS | ACTIONS |
|---|---|---|---|---|
| Dec 18, 2025 | 09:00 AM | Kyla | UNPAID | Details |
| Dec 18, 2025 | 11:00 AM | Yara | PAID | Details |
| Dec 18, 2025 | 01:00 PM | Jane | UNPAID | Details |

• Online (272 ms)

Medisked v1.0 | User: doctor1 | 2025-12-18 11:12:09 PM

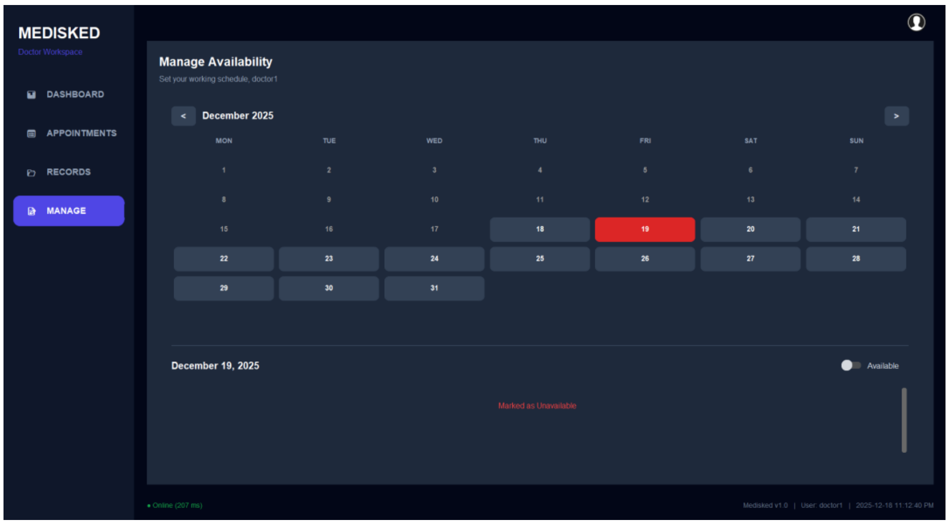


**My Records**

History of completed appointments for Dr. doctor1

| DATE | TIME | PATIENT | NOTES | ACTIONS |
|---|---|---|---|---|
| Dec 18, 2025 | 01:00 PM | Jane | Contact: 0909009093 \| Address: Davao City \| About:... | Details |
| Dec 18, 2025 | 11:00 AM | Yara | Contact: 984029043242 \| Address: Davao City \| Abou... | Details |
| Dec 18, 2025 | 09:00 AM | Kyla | Contact: 09002404024 \| Address: Davao City \| About... | Details |
| Dec 10, 2025 | 01:00 PM | Adreanah | Contact: 0986353261789 \| Address: Iwha, Toril \| Ab... | Details |
| Dec 10, 2025 | 11:00 AM | Kean | Contact: 0935472849567 \| Address: Matina Pangi, Da... | Details |
| Dec 10, 2025 | 09:00 AM | Orlandgee | Contact: 09453475869 \| Address: Coca Cola Village,... | Details |
| Dec 09, 2025 | 09:00 AM | Kate | Contact: 09092377974 \| Address: Matina Aplaya \| Ab... | Details |

• Online (186 ms)  Medisked v1.0 | User: doctor1 | 2025-12-18 11:12:22 PM



**Manage Availability**

Set your working schedule, doctor1

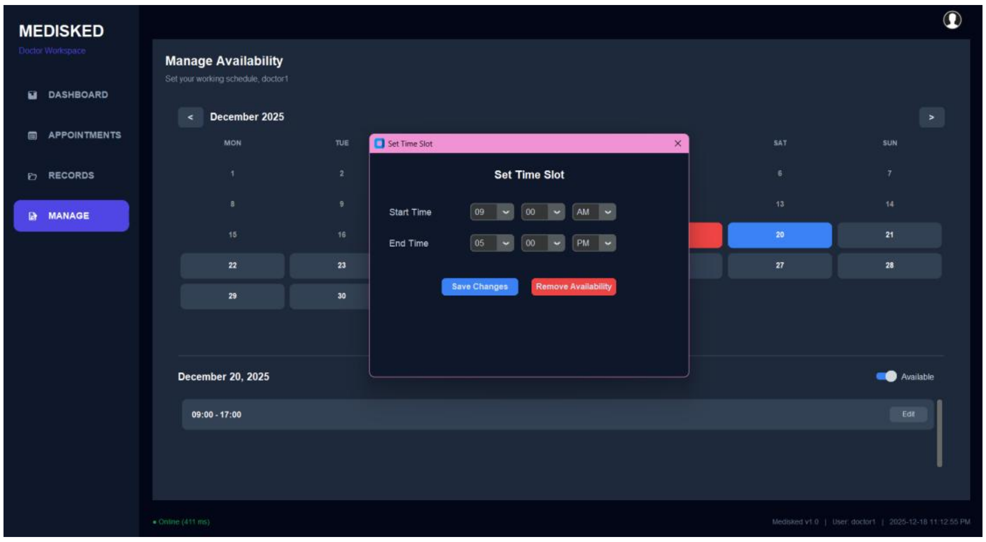
December 2025

| MON | TUE | WED | THU | FRI | SAT | SUN |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | | | | |

December 19, 2025

Marked as Unavailable

• Online (207 ms)  Medisked v1.0 | User: doctor1 | 2025-12-18 11:12:40 PM



**Manage Availability**

Set your working schedule, doctor1

December 2025

Set Time Slot

**Set Time Slot**

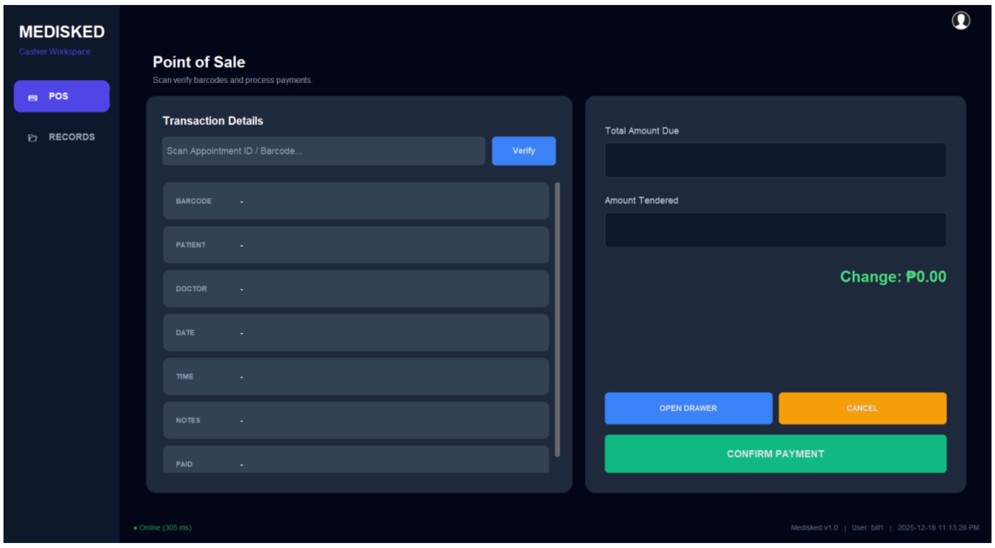
Start Time 09 00 AM

End Time 05 00 PM

December 20, 2025

09:00 - 17:00

• Online (411 ms)  Medisked v1.0 | User: doctor1 | 2025-12-18 11:12:55 PM



Point of Sale

Scan verify barcodes and process payments

Transaction Details

| | |
|---|---|
| BARCODE | - |
| PATIENT | - |
| DOCTOR | - |
| DATE | - |
| TIME | - |
| NOTES | - |
| PAID | - |

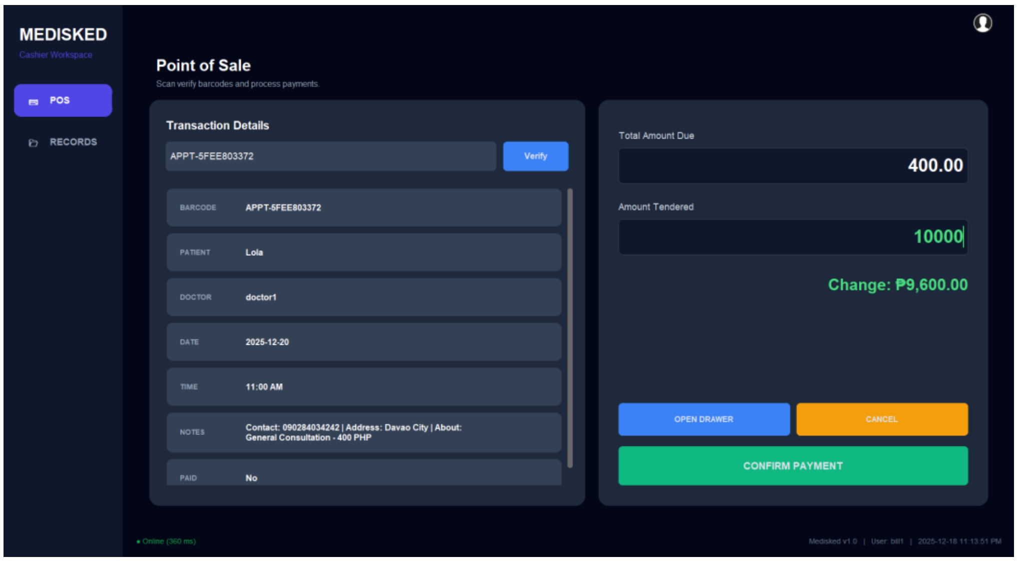
Total Amount Due

Amount Tendered

Change: ₱0.00

● Online (305 ms)

Medisked v1.0 | User: bill1 | 2025-12-18 11:13:28 PM



Point of Sale

Scan verify barcodes and process payments.

Transaction Details

APPT-5FEE803372

| | |
|---|---|
| BARCODE | APPT-5FEE803372 |
| PATIENT | Lola |
| DOCTOR | doctor1 |
| DATE | 2025-12-20 |
| TIME | 11:00 AM |
| NOTES | Contact: 090284034242 \| Address: Davao City \| About: General Consultation - 400 PHP |
| PAID | No |

Total Amount Due

400.00

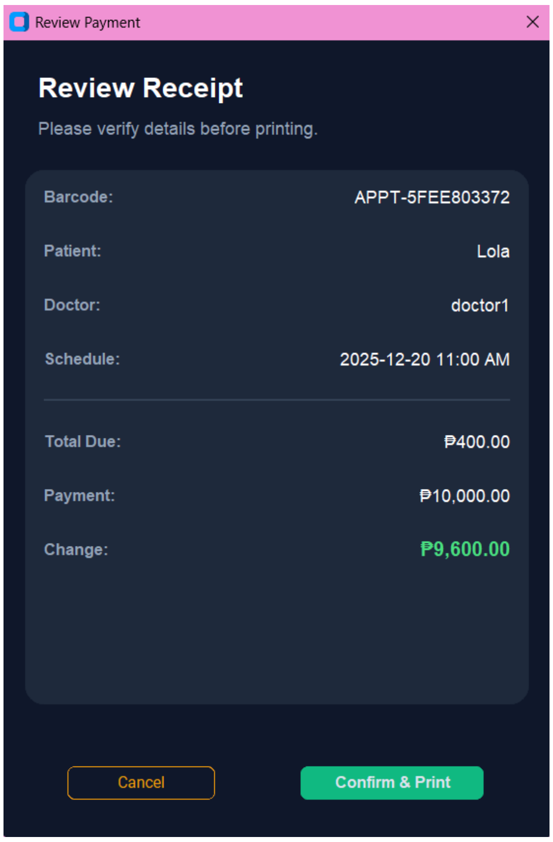
Amount Tendered

10000

Change: ₱9,600.00

● Online (360 ms)

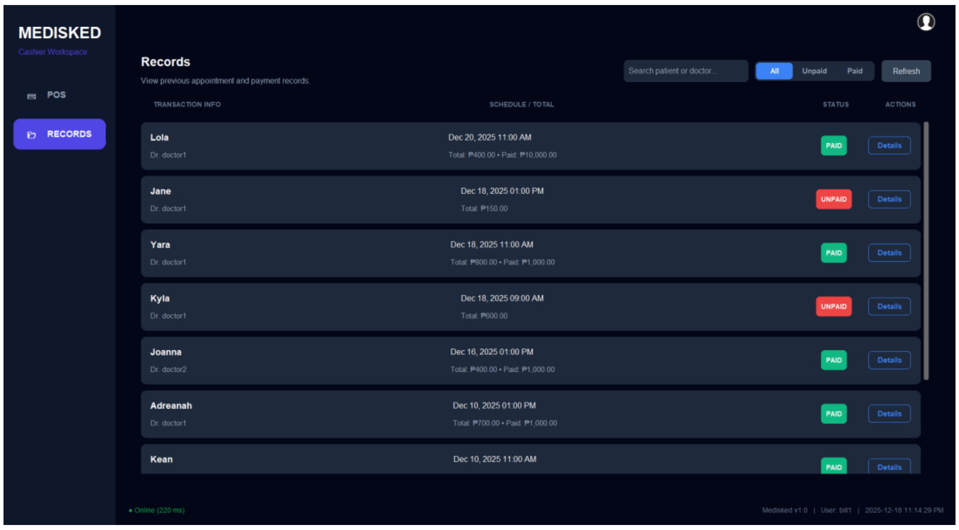
Medisked v1.0 | User: bill1 | 2025-12-18 11:13:51 PM



Review Receipt

Please verify details before printing.

| | |
|---|---|
| Barcode: | APPT-5FEE803372 |
| Patient: | Lola |
| Doctor: | doctor1 |
| Schedule: | 2025-12-20 11:00 AM |
| Total Due: | ₱400.00 |
| Payment: | ₱10,000.00 |
| Change: | ₱9,600.00 |



MEDISKED

Cashier Workspace

POS

RECORDS

Records

View previous appointment and payment records.

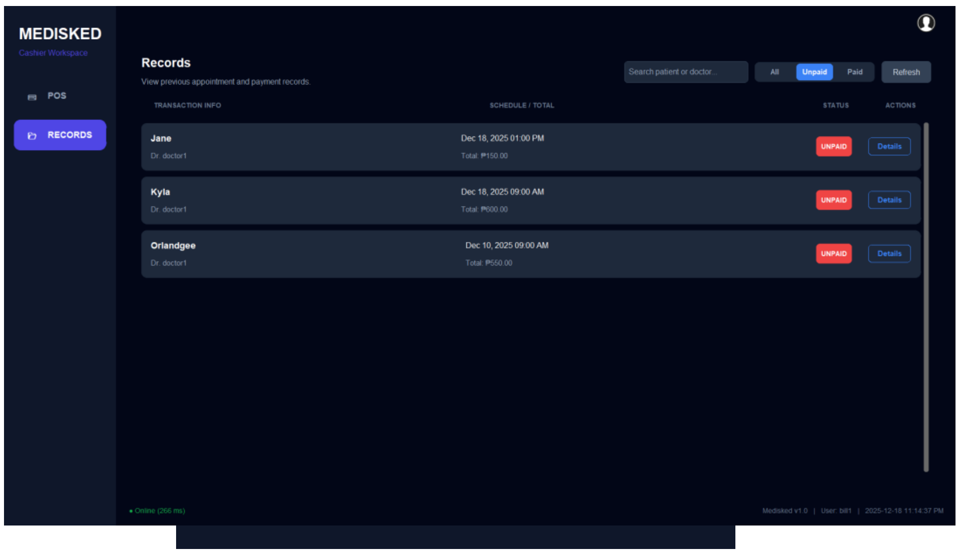
Search patient or doctor...

All | Unpaid | Paid | Refresh

| TRANSACTION INFO | SCHEDULE / TOTAL | STATUS | ACTIONS |
|---|---|---|---|
| Lola<br>Dr. doctor1 | Dec 20, 2025 11:00 AM<br>Total: ₱400.00 • Paid: ₱10,000.00 | PAID | Details |
| Jane<br>Dr. doctor1 | Dec 18, 2025 01:00 PM<br>Total: ₱150.00 | UNPAID | Details |
| Yara<br>Dr. doctor1 | Dec 18, 2025 11:00 AM<br>Total: ₱800.00 • Paid: ₱1,000.00 | PAID | Details |
| Kyla<br>Dr. doctor1 | Dec 18, 2025 09:00 AM<br>Total: ₱600.00 | UNPAID | Details |
| Joanna<br>Dr. doctor2 | Dec 16, 2025 01:00 PM<br>Total: ₱400.00 • Paid: ₱1,000.00 | PAID | Details |
| Adreanah<br>Dr. doctor1 | Dec 10, 2025 01:00 PM<br>Total: ₱700.00 • Paid: ₱1,000.00 | PAID | Details |
| Kean | Dec 10, 2025 11:00 AM | PAID | Details |

● Online (220 ms)

Medisked v1.0 | User: bill1 | 2025-12-18 11:14:29 PM



MEDISKED

Cashier Workspace

POS

RECORDS

Records

View previous appointment and payment records.

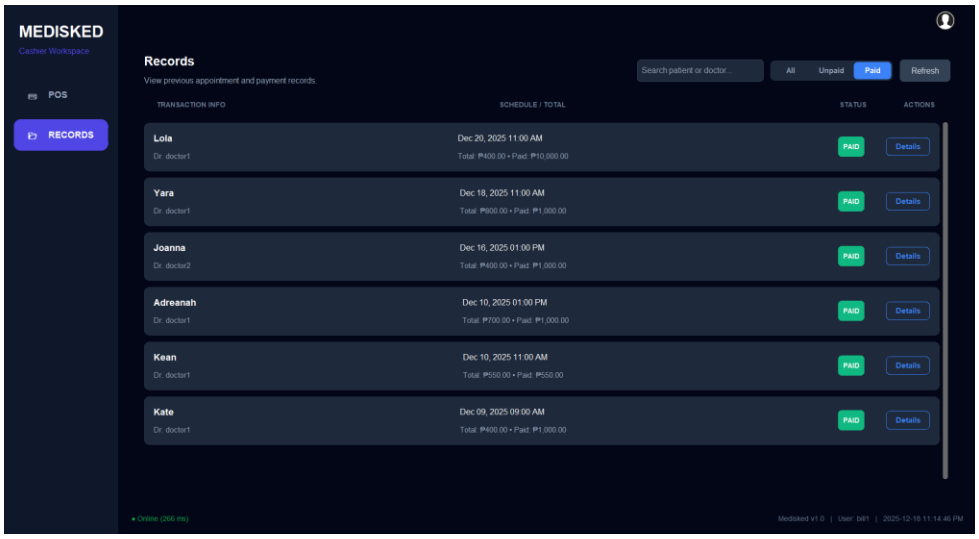
Search patient or doctor...

All | Unpaid | Paid | Refresh

| TRANSACTION INFO | SCHEDULE / TOTAL | STATUS | ACTIONS |
|---|---|---|---|
| Jane<br>Dr. doctor1 | Dec 18, 2025 01:00 PM<br>Total: ₱150.00 | UNPAID | Details |
| Kyla<br>Dr. doctor1 | Dec 18, 2025 09:00 AM<br>Total: ₱600.00 | UNPAID | Details |
| Orlandgee<br>Dr. doctor1 | Dec 10, 2025 09:00 AM<br>Total: ₱550.00 | UNPAID | Details |

● Online (266 ms)

Medisked v1.0 | User: bill1 | 2025-12-18 11:14:37 PM



Records

View previous appointment and payment records.

| TRANSACTION INFO | SCHEDULE / TOTAL | STATUS | ACTIONS |
|---|---|---|---|
| Lola<br>Dr. doctor1 | Dec 20, 2025 11:00 AM<br>Total: ₱400.00 • Paid: ₱10,000.00 | PAID | Details |
| Yara<br>Dr. doctor1 | Dec 18, 2025 11:00 AM<br>Total: ₱800.00 • Paid: ₱1,000.00 | PAID | Details |
| Joanna<br>Dr. doctor2 | Dec 16, 2025 01:00 PM<br>Total: ₱400.00 • Paid: ₱1,000.00 | PAID | Details |
| Adreanah<br>Dr. doctor1 | Dec 10, 2025 01:00 PM<br>Total: ₱700.00 • Paid: ₱1,000.00 | PAID | Details |
| Kean<br>Dr. doctor1 | Dec 10, 2025 11:00 AM<br>Total: ₱550.00 • Paid: ₱550.00 | PAID | Details |
| Kate<br>Dr. doctor1 | Dec 09, 2025 09:00 AM<br>Total: ₱400.00 • Paid: ₱1,000.00 | PAID | Details |

Medisked v1.0 | User: bill1 | 2025-12-18 11:14:46 PM

## VIII. Source Code Listing

### 1. MAIN PROGRAM

### main.py - Application Entry Point

```python
def main():
    """Main application loop with role-based routing"""
    while True:
        init_db(DB_NAME)

        # Login Phase
        login_app = LoginApp()
        login_app.mainloop()

        if not login_app.authenticated:
            return

        username = login_app.logged_in_user
        role = login_app.logged_in_role

        # Role-based Dashboard Loading
        if role == "admin":
            app = AdminDashboard(username=username)
        elif role == "doctor":
            app = DoctorDashboard(username=username)
        elif role == "cashier":
            from cashier_dashboard import CashierDashboard
            app = CashierDashboard(username=username)
        else:
            app = MainApp(username=username, role=role)

        app.mainloop()

        if not getattr(app, "should_relogin", False):
            break
```

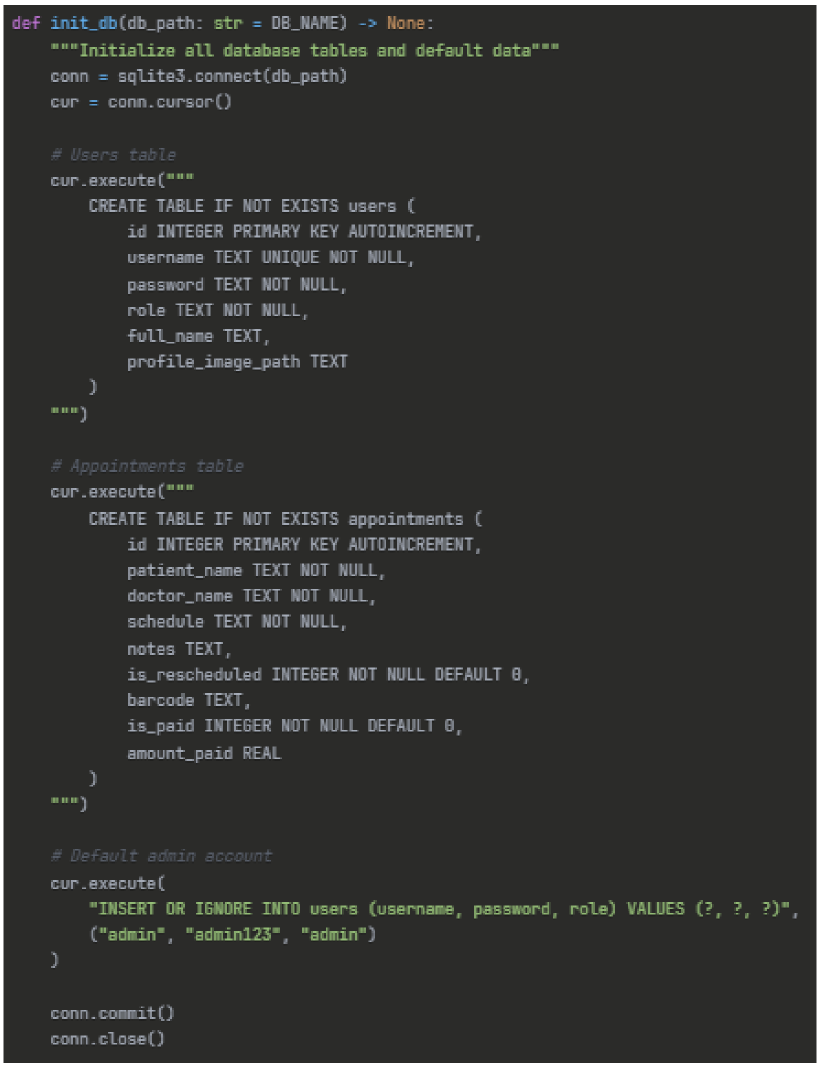
Key Features:

- Continuous login/logout loop
- Role-based dashboard initialization
- Database initialization on startup

## 2. CORE MODULES

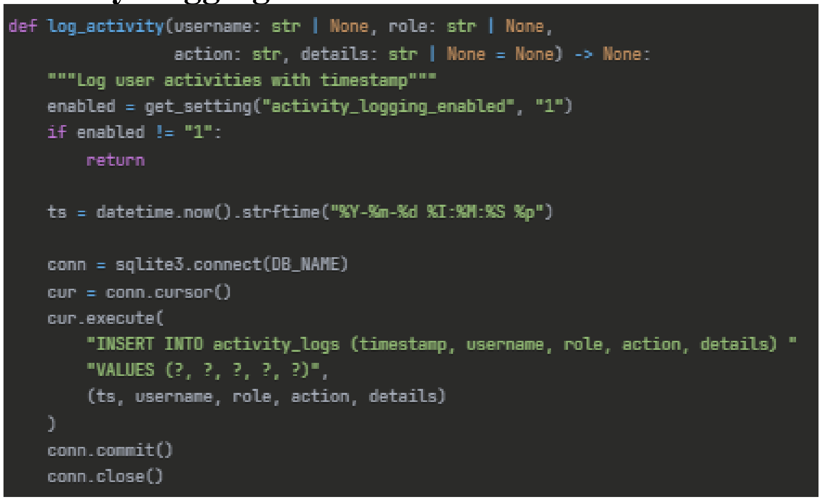
### database.py - Database Management Module

### Database Initialization



```python
def init_db(db_path: str = DB_NAME) -> None:
    """Initialize all database tables and default data"""
    conn = sqlite3.connect(db_path)
    cur = conn.cursor()

    # Users table
    cur.execute("""
        CREATE TABLE IF NOT EXISTS users (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            username TEXT UNIQUE NOT NULL,
            password TEXT NOT NULL,
            role TEXT NOT NULL,
            full_name TEXT,
            profile_image_path TEXT
        )
    """)

    # Appointments table
    cur.execute("""
        CREATE TABLE IF NOT EXISTS appointments (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            patient_name TEXT NOT NULL,
            doctor_name TEXT NOT NULL,
            schedule TEXT NOT NULL,
            notes TEXT,
            is_rescheduled INTEGER NOT NULL DEFAULT 0,
            barcode TEXT,
            is_paid INTEGER NOT NULL DEFAULT 0,
            amount_paid REAL
        )
    """)

    # Default admin account
    cur.execute(
        "INSERT OR IGNORE INTO users (username, password, role) VALUES (?, ?, ?)",
        ("admin", "admin123", "admin")
    )

    conn.commit()
    conn.close()
```

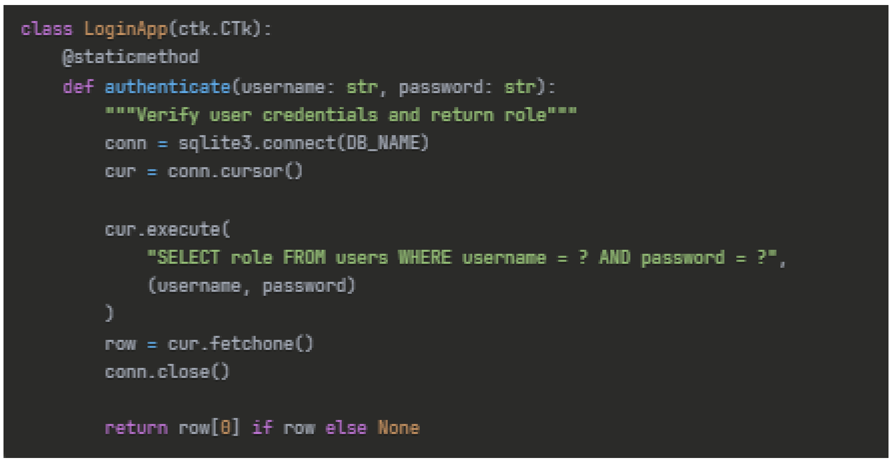
## Activity Logging



```python
def log_activity(username: str | None, role: str | None,
                 action: str, details: str | None = None) -> None:
    """Log user activities with timestamp"""
    enabled = get_setting("activity_logging_enabled", "1")
    if enabled != "1":
        return

    ts = datetime.now().strftime("%Y-%m-%d %I:%M:%S %p")

    conn = sqlite3.connect(DB_NAME)
    cur = conn.cursor()
    cur.execute(
        "INSERT INTO activity_logs (timestamp, username, role, action, details) "
        "VALUES (?, ?, ?, ?, ?)",
        (ts, username, role, action, details)
    )
    conn.commit()
    conn.close()
```

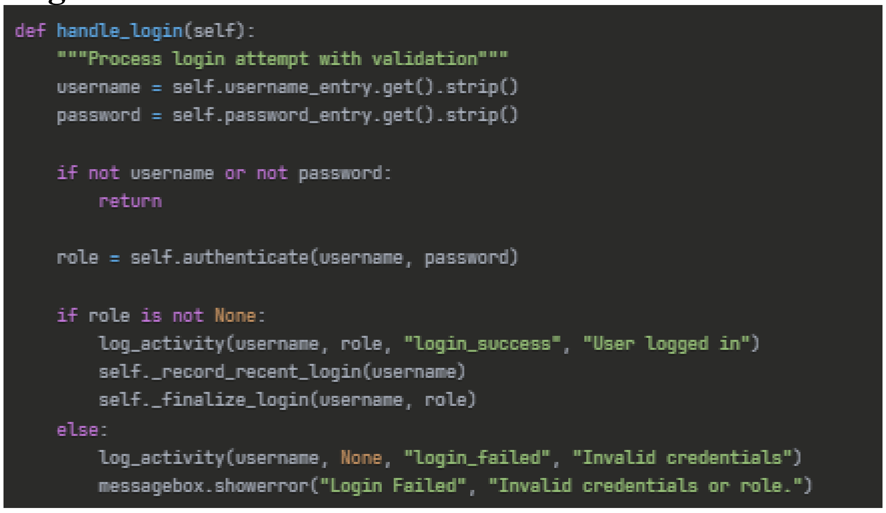
## login.py - Authentication Module

### User Authentication



```python
class LoginApp(ctk.CTk):
    @staticmethod
    def authenticate(username: str, password: str):
        """Verify user credentials and return role"""
        conn = sqlite3.connect(DB_NAME)
        cur = conn.cursor()

        cur.execute(
            "SELECT role FROM users WHERE username = ? AND password = ?",
            (username, password)
        )
        row = cur.fetchone()
        conn.close()

        return row[0] if row else None
```

## Login Handler

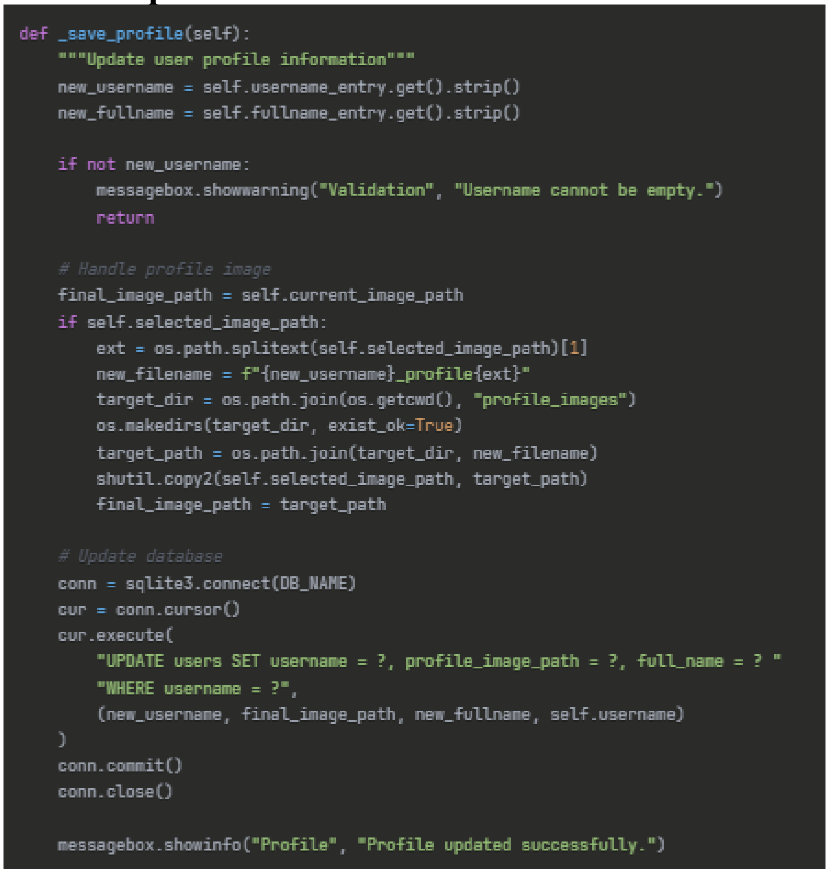


```python
def handle_login(self):
    """Process login attempt with validation"""
    username = self.username_entry.get().strip()
    password = self.password_entry.get().strip()

    if not username or not password:
        return

    role = self.authenticate(username, password)

    if role is not None:
        log_activity(username, role, "login_success", "User logged in")
        self._record_recent_login(username)
        self._finalize_login(username, role)
    else:
        log_activity(username, None, "login_failed", "Invalid credentials")
        messagebox.showerror("Login Failed", "Invalid credentials or role.")
```

## 3. KEY FUNCTIONS

### Profile Management (profile_window.py)

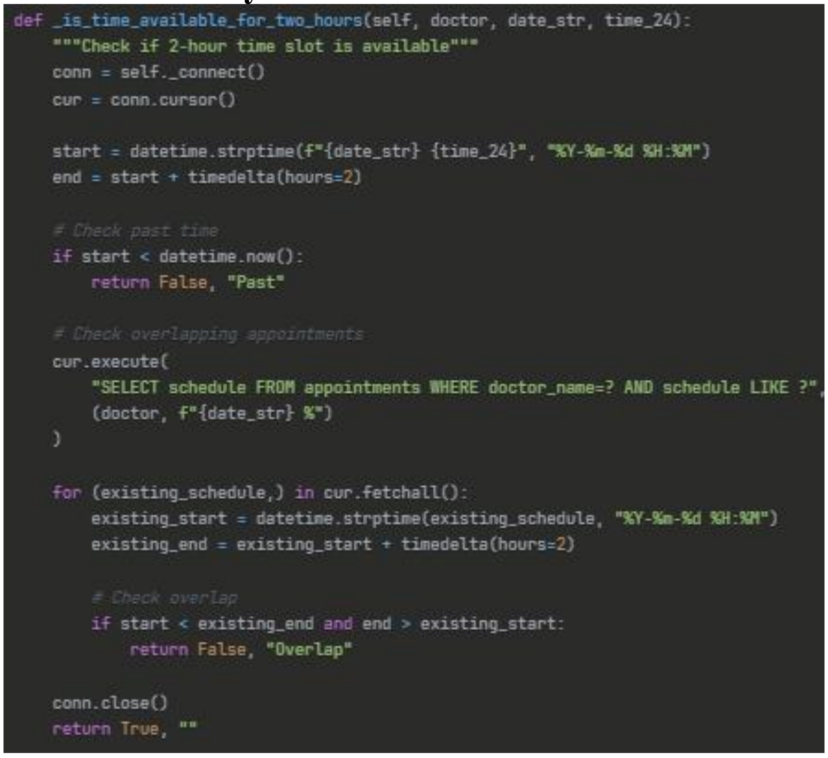
## Profile Update Function



```python
def _save_profile(self):
    """Update user profile information"""
    new_username = self.username_entry.get().strip()
    new_fullname = self.fullname_entry.get().strip()

    if not new_username:
        messagebox.showwarning("Validation", "Username cannot be empty.")
        return

    # Handle profile image
    final_image_path = self.current_image_path
    if self.selected_image_path:
        ext = os.path.splitext(self.selected_image_path)[1]
        new_filename = f"{new_username}_profile{ext}"
        target_dir = os.path.join(os.getcwd(), "profile_images")
        os.makedirs(target_dir, exist_ok=True)
        target_path = os.path.join(target_dir, new_filename)
        shutil.copy2(self.selected_image_path, target_path)
        final_image_path = target_path

    # Update database
    conn = sqlite3.connect(DB_NAME)
    cur = conn.cursor()
    cur.execute(
        "UPDATE users SET username = ?, profile_image_path = ?, full_name = ? "
        "WHERE username = ?",
        (new_username, final_image_path, new_fullname, self.username)
    )
    conn.commit()
    conn.close()

    messagebox.showinfo("Profile", "Profile updated successfully.")
```

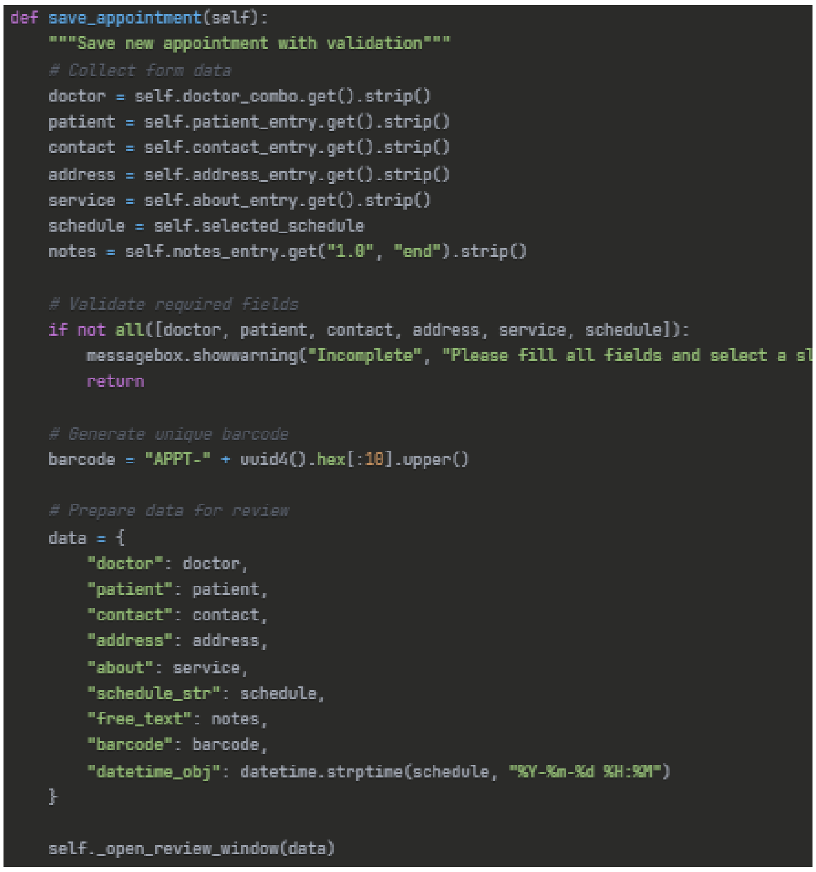
## Appointment Booking (receptionist_appointment_page.py)

### Slot Availability Check



```python
def _is_time_available_for_two_hours(self, doctor, date_str, time_24):
    """Check if 2-hour time slot is available"""
    conn = self._connect()
    cur = conn.cursor()

    start = datetime.strptime(f"{date_str} {time_24}", "%Y-%m-%d %H:%M")
    end = start + timedelta(hours=2)

    # Check past time
    if start < datetime.now():
        return False, "Past"

    # Check overlapping appointments
    cur.execute(
        "SELECT schedule FROM appointments WHERE doctor_name=? AND schedule LIKE ?",
        (doctor, f"{date_str} %")
    )

    for (existing_schedule,) in cur.fetchall():
        existing_start = datetime.strptime(existing_schedule, "%Y-%m-%d %H:%M")
        existing_end = existing_start + timedelta(hours=2)

        # Check overlap
        if start < existing_end and end > existing_start:
            return False, "Overlap"

    conn.close()
    return True, ""
```

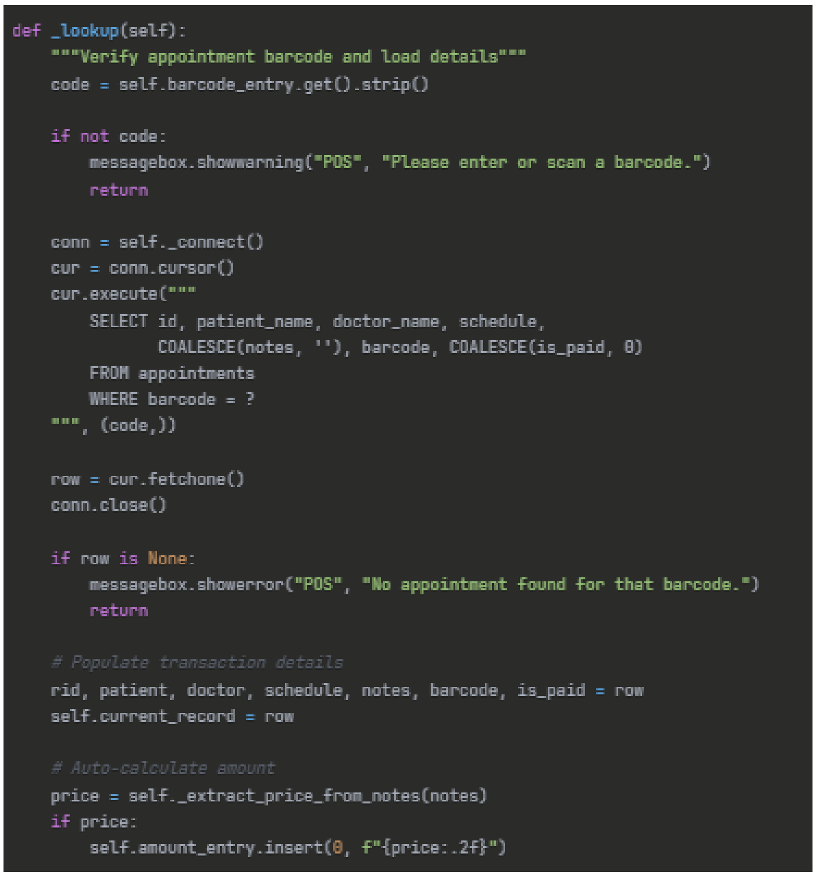
### Save Appointment



```python
def save_appointment(self):
    """Save new appointment with validation"""
    # Collect form data
    doctor = self.doctor_combo.get().strip()
    patient = self.patient_entry.get().strip()
    contact = self.contact_entry.get().strip()
    address = self.address_entry.get().strip()
    service = self.about_entry.get().strip()
    schedule = self.selected_schedule
    notes = self.notes_entry.get("1.0", "end").strip()

    # Validate required fields
    if not all([doctor, patient, contact, address, service, schedule]):
        messagebox.showwarning("Incomplete", "Please fill all fields and select a sl
        return

    # Generate unique barcode
    barcode = "APPT-" + uuid4().hex[:10].upper()

    # Prepare data for review
    data = {
        "doctor": doctor,
        "patient": patient,
        "contact": contact,
        "address": address,
        "about": service,
        "schedule_str": schedule,
        "free_text": notes,
        "barcode": barcode,
        "datetime_obj": datetime.strptime(schedule, "%Y-%m-%d %H:%M")
    }

    self._open_review_window(data)
```

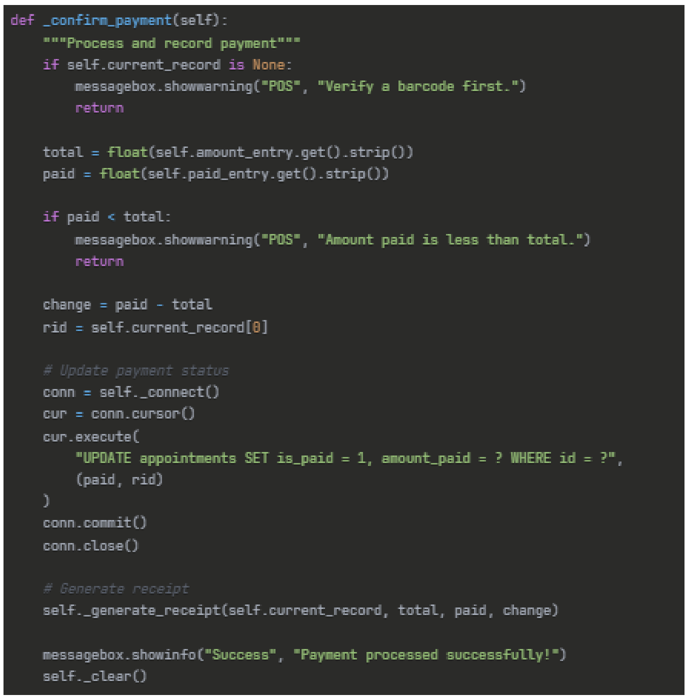
## Payment Processing (cashier_pos_page.py)

### Barcode Verification



```python
def _lookup(self):
    """Verify appointment barcode and load details"""
    code = self.barcode_entry.get().strip()

    if not code:
        messagebox.showwarning("POS", "Please enter or scan a barcode.")
        return

    conn = self._connect()
    cur = conn.cursor()
    cur.execute("""
        SELECT id, patient_name, doctor_name, schedule,
               COALESCE(notes, ''), barcode, COALESCE(is_paid, 0)
        FROM appointments
        WHERE barcode = ?
    """, (code,))

    row = cur.fetchone()
    conn.close()

    if row is None:
        messagebox.showerror("POS", "No appointment found for that barcode.")
        return

    # Populate transaction details
    rid, patient, doctor, schedule, notes, barcode, is_paid = row
    self.current_record = row

    # Auto-calculate amount
    price = self._extract_price_from_notes(notes)
    if price:
        self.amount_entry.insert(0, f"{price:.2f}")
```

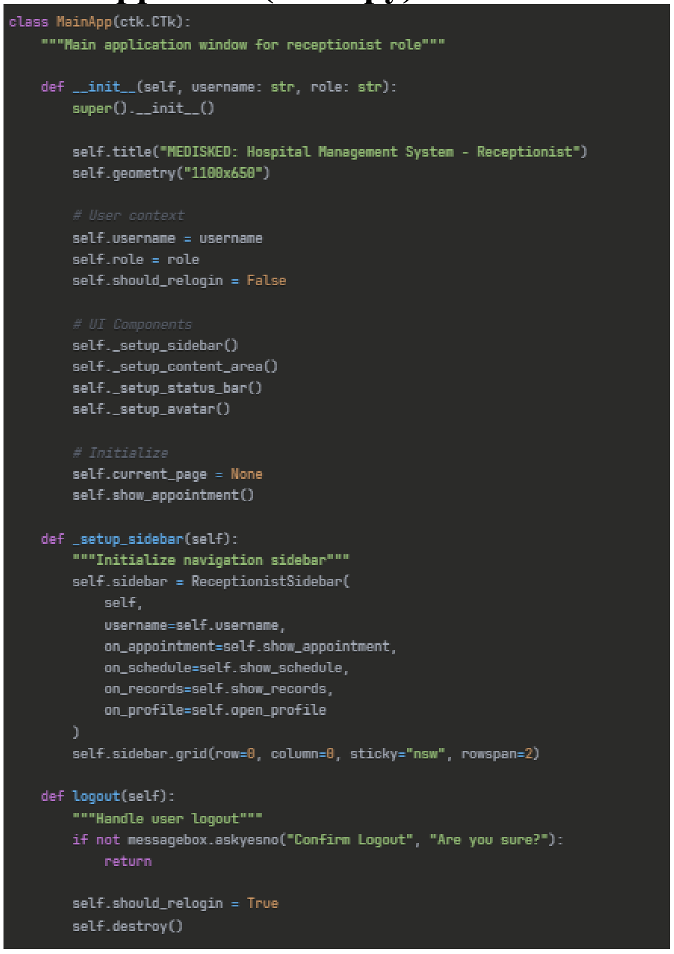
### Payment Confirmation



```python
def _confirm_payment(self):
    """Process and record payment"""
    if self.current_record is None:
        messagebox.showwarning("POS", "Verify a barcode first.")
        return

    total = float(self.amount_entry.get().strip())
    paid = float(self.paid_entry.get().strip())

    if paid < total:
        messagebox.showwarning("POS", "Amount paid is less than total.")
        return

    change = paid - total
    rid = self.current_record[0]

    # Update payment status
    conn = self._connect()
    cur = conn.cursor()
    cur.execute(
        "UPDATE appointments SET is_paid = 1, amount_paid = ? WHERE id = ?",
        (paid, rid)
    )
    conn.commit()
    conn.close()

    # Generate receipt
    self._generate_receipt(self.current_record, total, paid, change)

    messagebox.showinfo("Success", "Payment processed successfully!")
    self._clear()
```

## 4. CLASS DEFINITIONS

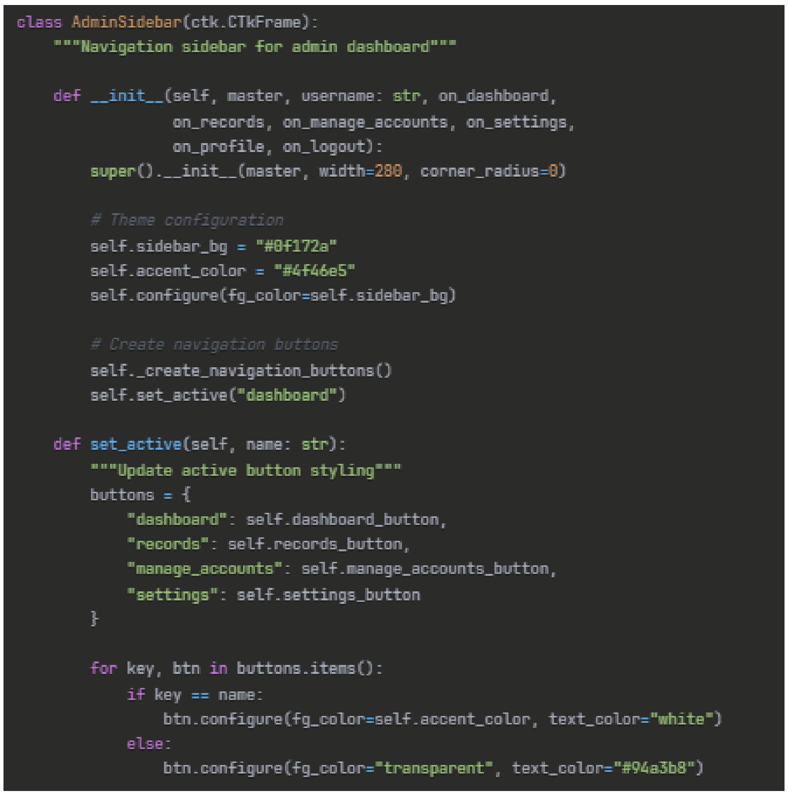
### MainApp Class (main.py)



```python
class MainApp(ctk.CTk):
    """Main application window for receptionist role"""

    def __init__(self, username: str, role: str):
        super().__init__()

        self.title("MEDISKED: Hospital Management System - Receptionist")
        self.geometry("1100x650")

        # User context
        self.username = username
        self.role = role
        self.should_relogin = False

        # UI Components
        self._setup_sidebar()
        self._setup_content_area()
        self._setup_status_bar()
        self._setup_avatar()

        # Initialize
        self.current_page = None
        self.show_appointment()

    def _setup_sidebar(self):
        """Initialize navigation sidebar"""
        self.sidebar = ReceptionistSidebar(
            self,
            username=self.username,
            on_appointment=self.show_appointment,
            on_schedule=self.show_schedule,
            on_records=self.show_records,
            on_profile=self.open_profile
        )
        self.sidebar.grid(row=0, column=0, sticky="nsw", rowspan=2)

    def logout(self):
        """Handle user logout"""
        if not messagebox.askyesno("Confirm Logout", "Are you sure?"):
            return

        self.should_relogin = True
        self.destroy()
```

### Sidebar Classes

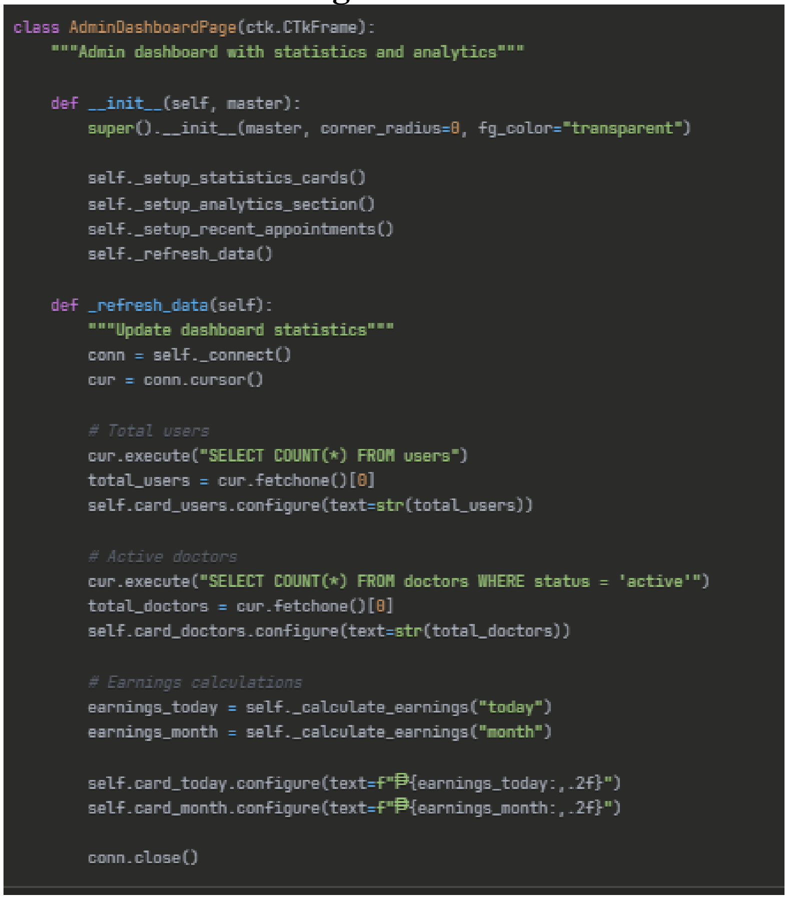
## AdminSidebar



```python
class AdminSidebar(ctk.CTkFrame):
    """Navigation sidebar for admin dashboard"""

    def __init__(self, master, username: str, on_dashboard,
                 on_records, on_manage_accounts, on_settings,
                 on_profile, on_logout):
        super().__init__(master, width=280, corner_radius=0)

        # Theme configuration
        self.sidebar_bg = "#0f172a"
        self.accent_color = "#4f46e5"
        self.configure(fg_color=self.sidebar_bg)

        # Create navigation buttons
        self._create_navigation_buttons()
        self.set_active("dashboard")

    def set_active(self, name: str):
        """Update active button styling"""
        buttons = {
            "dashboard": self.dashboard_button,
            "records": self.records_button,
            "manage_accounts": self.manage_accounts_button,
            "settings": self.settings_button
        }

        for key, btn in buttons.items():
            if key == name:
                btn.configure(fg_color=self.accent_color, text_color="white")
            else:
                btn.configure(fg_color="transparent", text_color="#94a3b8")
```

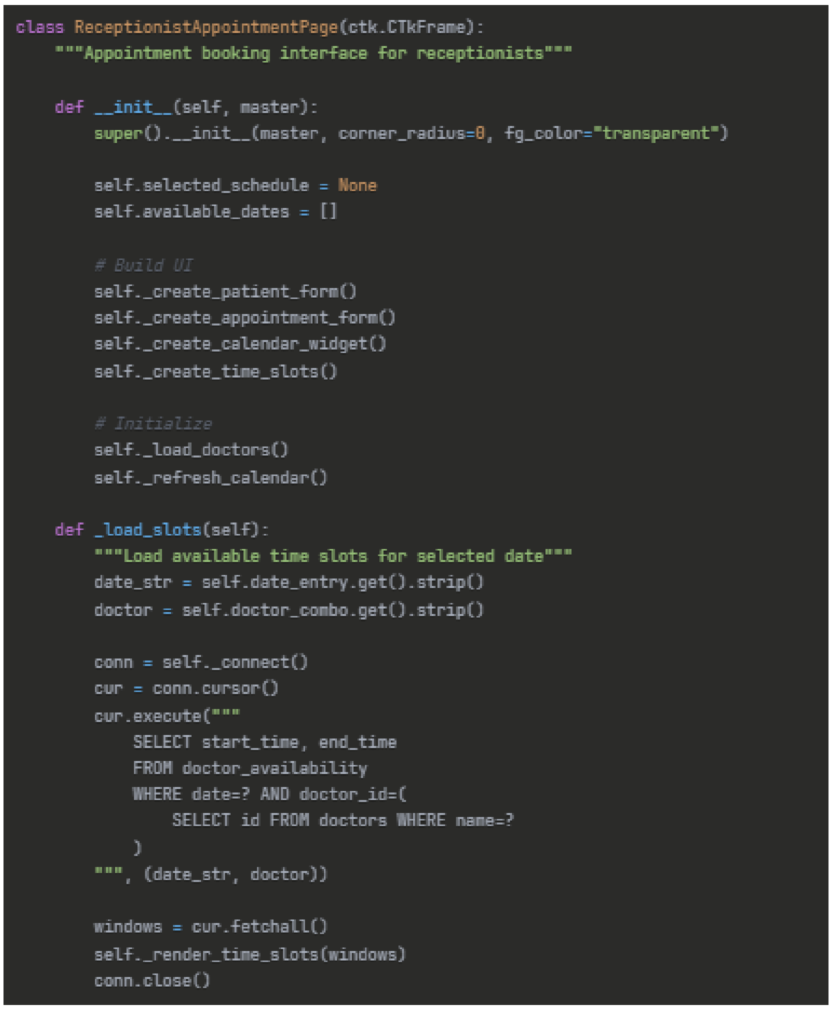
## Page Classes

### AdminDashboardPage



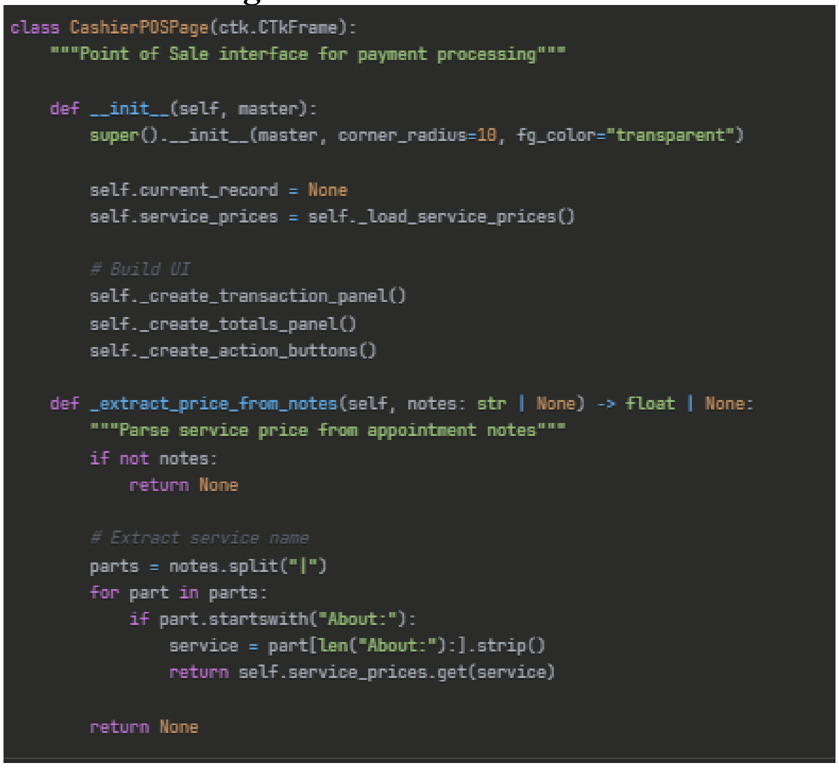
```python
class AdminDashboardPage(ctk.CTkFrame):
    """Admin dashboard with statistics and analytics"""

    def __init__(self, master):
        super().__init__(master, corner_radius=0, fg_color="transparent")

        self._setup_statistics_cards()
        self._setup_analytics_section()
        self._setup_recent_appointments()
        self._refresh_data()

    def _refresh_data(self):
        """Update dashboard statistics"""
        conn = self._connect()
        cur = conn.cursor()

        # Total users
        cur.execute("SELECT COUNT(*) FROM users")
        total_users = cur.fetchone()[0]
        self.card_users.configure(text=str(total_users))

        # Active doctors
        cur.execute("SELECT COUNT(*) FROM doctors WHERE status = 'active'")
        total_doctors = cur.fetchone()[0]
        self.card_doctors.configure(text=str(total_doctors))

        # Earnings calculations
        earnings_today = self._calculate_earnings("today")
        earnings_month = self._calculate_earnings("month")

        self.card_today.configure(text=f"₱{earnings_today:,.2f}")
        self.card_month.configure(text=f"₱{earnings_month:,.2f}")

        conn.close()
```

### ReceptionistAppointmentPage



```python
class ReceptionistAppointmentPage(ctk.CTkFrame):
    """Appointment booking interface for receptionists"""

    def __init__(self, master):
        super().__init__(master, corner_radius=0, fg_color="transparent")

        self.selected_schedule = None
        self.available_dates = []

        # Build UI
        self._create_patient_form()
        self._create_appointment_form()
        self._create_calendar_widget()
        self._create_time_slots()

        # Initialize
        self._load_doctors()
        self._refresh_calendar()

    def _load_slots(self):
        """Load available time slots for selected date"""
        date_str = self.date_entry.get().strip()
        doctor = self.doctor_combo.get().strip()

        conn = self._connect()
        cur = conn.cursor()
        cur.execute("""
            SELECT start_time, end_time
            FROM doctor_availability
            WHERE date=? AND doctor_id=(
                SELECT id FROM doctors WHERE name=?
            )
        """, (date_str, doctor))

        windows = cur.fetchall()
        self._render_time_slots(windows)
        conn.close()
```

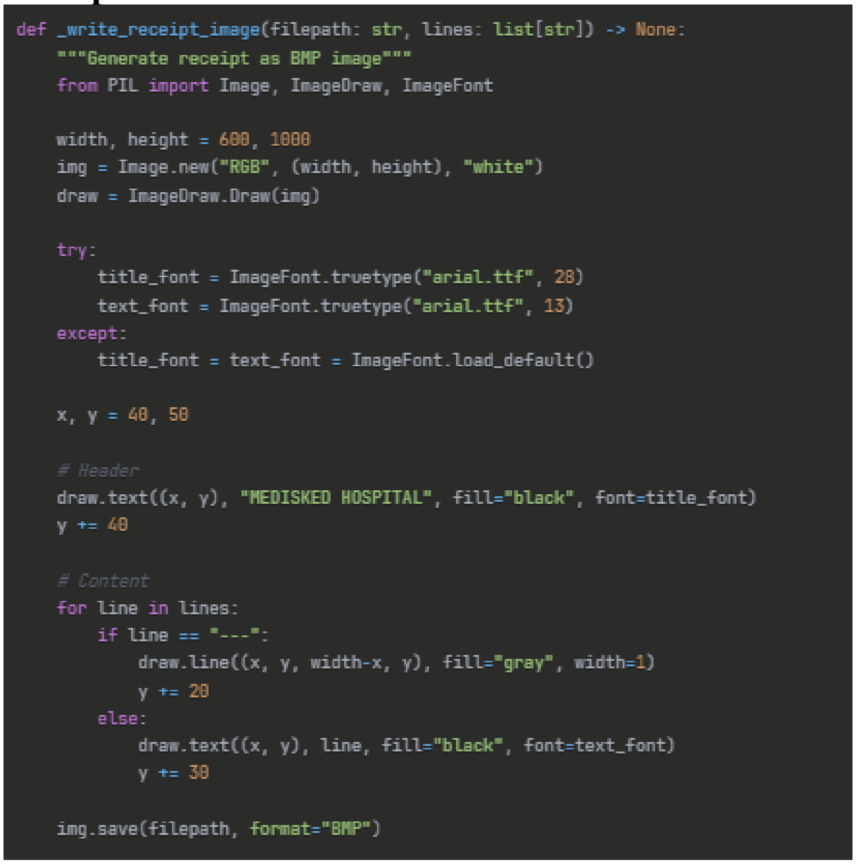
## CashierPOSPage



```python
class CashierPOSPage(ctk.CTkFrame):
    """Point of Sale interface for payment processing"""

    def __init__(self, master):
        super().__init__(master, corner_radius=10, fg_color="transparent")

        self.current_record = None
        self.service_prices = self._load_service_prices()

        # Build UI
        self._create_transaction_panel()
        self._create_totals_panel()
        self._create_action_buttons()

    def _extract_price_from_notes(self, notes: str | None) -> float | None:
        """Parse service price from appointment notes"""
        if not notes:
            return None

        # Extract service name
        parts = notes.split("|")
        for part in parts:
            if part.startswith("About:"):
                service = part[len("About:"):].strip()
                return self.service_prices.get(service)

        return None
```

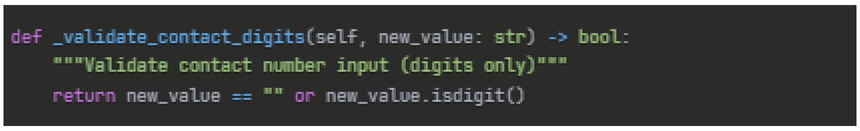
## 6. UTILITY FUNCTIONS

### Receipt Generation



```python
def _write_receipt_image(filepath: str, lines: list[str]) -> None:
    """Generate receipt as BMP image"""
    from PIL import Image, ImageDraw, ImageFont

    width, height = 600, 1000
    img = Image.new("RGB", (width, height), "white")
    draw = ImageDraw.Draw(img)

    try:
        title_font = ImageFont.truetype("arial.ttf", 28)
        text_font = ImageFont.truetype("arial.ttf", 13)
    except:
        title_font = text_font = ImageFont.load_default()

    x, y = 40, 50

    # Header
    draw.text((x, y), "MEDISKED HOSPITAL", fill="black", font=title_font)
    y += 40

    # Content
    for line in lines:
        if line == "---":
            draw.line((x, y, width-x, y), fill="gray", width=1)
            y += 20
        else:
            draw.text((x, y), line, fill="black", font=text_font)
            y += 30

    img.save(filepath, format="BMP")
```

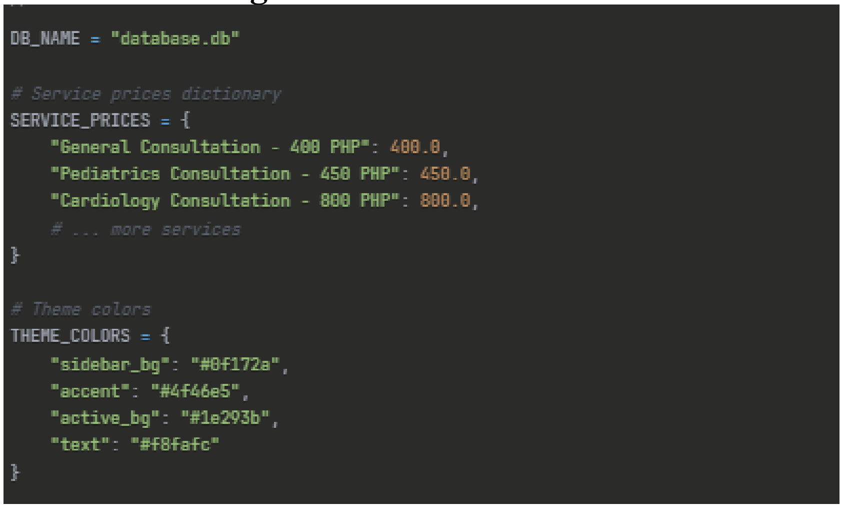
### Contact Validation



```python
def _validate_contact_digits(self, new_value: str) -> bool:
    """Validate contact number input (digits only)"""
    return new_value == "" or new_value.isdigit()
```

## 6. CONFIGURATION & CONSTANTS

### Database Configuration



```python
DB_NAME = "database.db"

# Service prices dictionary
SERVICE_PRICES = {
    "General Consultation - 400 PHP": 400.0,
    "Pediatrics Consultation - 450 PHP": 450.0,
    "Cardiology Consultation - 800 PHP": 800.0,
    # ... more services
}

# Theme colors
THEME_COLORS = {
    "sidebar_bg": "#0f172a",
    "accent": "#4f46e5",
    "active_bg": "#1e293b",
    "text": "#f8fafc"
}
```

## IX. Testing & Results

**Test Case Overview**
**Project Name:** MediSked - Hospital Scheduling and Billing Management System
**Version:** 1.0
**Test Date:** December 2025
**Test Environment:** Windows 10/11, Python 3.x, SQLite Database

**1. LOGIN MODULE TEST CASES**

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|
| TC-001: Successful Admin Login | Username: admin<br>Password: admin123 | -"Verifying credentials..." message<br>-"Login Successful - Welcome, admin"<br>-Redirect to Admin Dashboard | "Verifying credentials..." message<br>"Login Successful - Welcome, admin"<br>Redirect to Admin Dashboard | Passed |
| TC-002: Failed Login - Invalid Username | Username: invaliduser<br>Password: password123 | -Error message: "Invalid credentials or role"<br>-Remain on login screen<br>-No access granted | Error message: "Invalid credentials or role"<br>Remain on login screen<br>No access granted | Passed |
| TC-003: Failed Login - Invalid Password | Username: admin<br>Password: wrongpassword | -Error message: "Invalid credentials or role"<br>-Remain on login screen<br>-No access granted | Error message: "Invalid credentials or role"<br>Remain on login screen<br>No access granted | Passed |
| TC-004: Login - Empty Fields | Username: (empty)<br>Password: (empty) | -Validation error message<br>-Login prevented<br>-Remain on login screen | Validation error message<br>Login prevented<br>Remain on login screen | Passed |
| TC-005: Password Visibility Toggle | Password: admin123<br>Click eye icon | -Password shown as asterisks initially<br>-Becomes | Password shown as asterisks initially<br>Becomes | Passed |

| | | | | |
|---|---|---|---|---|
| | | visible when eye clicked -Returns to asterisks when clicked again | visible when eye clicked Returns to asterisks when clicked again | |
| TC-006: Receptionist Role Login | Username: reception1 Password: [created password] | -"Login Successful - Welcome, reception1" -Redirect to Receptionist dashboard -Only receptionist menu visible | "Verifying credentials..." "Login Successful - Welcome, reception1" Redirect to Receptionist dashboard Only receptionist menu visible | Passed |

## 2. ADMIN MODULE TEST CASES

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|
| TC-007: Create Doctor Account | Username: doctor3 Password: doctor123 Role: doctor | -"Doctor account created" confirmation -Account saved to database -Appears in Users list | "Doctor account created" confirmation Account saved to database Appears in Users list | Passed |
| TC-008: Create Receptionist Account | Username: reception2 Password: recep123 Role: receptionist | -"Receptionist account created" confirmation -Account saved to database -Visible in Users list | "Receptionist account created" confirmation Account saved to database Visible in Users list | Passed |
| TC-009: Create Cashier Account | Username: bill2 Password: bill123 Role: cashier | -"Cashier account created" confirmation -Account saved to database -Visible in Users list | "Cashier account created" confirmation Account saved to database Visible in Users list | Passed |
| TC-010: View Admin Dashboard | Navigate to Dashboard | -Users count displayed -Doctors count shown | Users count displayed Doctors count shown | Passed |

|  |  |  |  |  |
|---|---|---|---|---|
|  |  | -Total appointments displayed<br>-Earnings today displayed<br>-Earnings this month displayed<br>-Average earnings calculated | Total appointments displayed<br>Earnings today displayed<br>Earnings this month displayed<br>Average earnings calculated |  |
| TC-011: View All Appointments | Navigate to Records | -All appointments displayed<br>-Shows: patient, doctor, date, time, status<br>-Search, Refresh, Export buttons functional | All appointments displayed<br>Shows: patient, doctor, date, time, status<br>Search, Refresh, Export buttons functional | Passed |
| TC-012: Export to CSV | Click "Export CSV" button | -"Appointments exported successfully"<br>-CSV file created with all records<br>-Correct format and readable<br>-Data matches system | "Appointments exported successfully"<br>CSV file created with all records<br>Correct format and readable<br>Data matches system | Passed |
| TC-013: View Activity Logs | Navigate to Settings > Activity Logs | -Logs displayed chronologically<br>-Shows: date, time, user, action, description<br>-Date filter works<br>-Refresh updates list | Logs displayed chronologically<br>Shows: date, time, user, action, description<br>Date filter works<br>Refresh updates list | Passed |

## 3. RECEPTIONIST MODULE TEST CASES

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| TC-014: Create New Appointment | Name: Jomarl<br>Contact: 09065629387<br>Address: prkajbdda<br>Service: Cardiology - 800 PHP<br>Doctor: doctor2<br>Date: 2025-12-20<br>Time: 01:00 PM | -Appointment review modal appears<br>-Barcode generated (APPT-XXXXXXXXXX)<br>-"Appointment created successfully"<br>-Saved with UNPAID status | Appointment review modal appears<br>Barcode generated (APPT-D52BBC9930)<br>"Appointment created successfully<br>Saved with UNPAID status | Passed |
| TC-015: Prevent Double Booking | Doctor: doctor2<br>Date: 2025-12-20<br>Time: 01:00 PM (already booked) | -Time slot grayed out/unavailable<br>-Selection prevented<br>-Alternative slots shown | Time slot grayed out/unavailable<br>Selection prevented<br>Alternative slots shown | Passed |
| TC-016: View Schedule Calendar | Select doctor from dropdown<br>View calendar | -Calendar with color-coded dates<br>-Green = available<br>-Red = unavailable<br>-Gray = time passed<br>-Click date shows time slots | Calendar with color-coded dates<br>Green = available<br>Red = unavailable<br>Gray = time passed<br>Click date shows time slots | Passed |
| TC-017: Search Patient Records | Search box: "Kate" | -Real-time filter<br>- Only matching appointments shown<br>-View/Edit buttons functional<br>-Clear resets search | Real-time filter<br>Only matching appointments shown (Kate)<br>View/Edit buttons functional<br>Clear and referesh resets search | Passed |
| TC-018: Edit Appointment | Click Edit on appointment<br>Modify contact number<br>Save | -Edit form opens with existing data<br>-Changes can be made<br>-Updated info saved | Edit form opens with existing data<br>Changes can be made<br>Updated info saved | Passed<br><br>Passed |

|  |  |  |  |  |
|---|---|---|---|---|
| TC-019: Logout Confirmation | Click user icon > Logout Click "Yes" | -"Are you sure?" confirmation -"Yes" logs out - "No" cancels -Return to login screen | "Are you sure?" confirmation "Yes" logs out "No" cancels Return to login screen | Passed |
| TC-020: Calendar Navigation | Click forward/backward arrows | -Forward advances month -Backward returns to previous -Month/year updates correctly -Availability reflects correct month | Forward advances month Backward returns to previous Month/year updates correctly Availability reflects correct month | Passed |

## 4. CASHIER MODULE TEST CASES

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|
| TC-021: Verify Appointment | Barcode: APPT-6DFF903092 Click "Verify" | -Details displayed: Patient, Doctor, Date, Time -Total Due: ₱700.00 - Paid: No -Amount field ready | Details displayed: Patient, Doctor, Date, Time -Total Due: ₱700.00 Paid: No Amount field ready | Passed |
| TC-022: Payment - Exact Amount | Amount: 700 Click "PROCEED" Confirm Print | -Change: ₱0.00 -Receipt modal shows all details -"Payment Successful & Receipt Printed!" -Status updated to PAID | Change: ₱0.00 Receipt modal shows all details "Payment Successful & Receipt Printed!" Status: PAID | Passed |
| TC-023: Payment - With Change | Total: ₱700.00 Amount: 7007 Click | -Change: ₱6,307.00 (green) | Change: ₱6,307.00 (green) | Passed |

| | | | | |
|---|---|---|---|---|
| | "PROCEED" | -Receipt shows correct amount<br>-Payment recorded<br>-Status: PAID | Review Receipt "Payment Successful & Receipt Printed!"<br>Status: PAID | |
| TC-024: Insufficient Amount | Total: ₱700.00<br>Amount: 500 | -Error: "Insufficient payment amount"<br>-PROCEED disabled<br>-Payment not processed<br>-Remains UNPAID | Error: "amount paid is less than total."<br>-PROCEED disabled<br>-Payment not processed<br>-Remains UNPAID | Passed |
| TC-025: Cancel Transaction | Verify appointment<br>Enter amount<br>Click "CANCEL" | -All fields cleared<br>-Amount reset to 0<br>-Change: ₱0.00<br>-Ready for new transaction | All fields cleared<br>Amount reset to 0<br>Change: ₱0.00<br>Ready for new transaction | Passed |
| TC-026: Invalid Barcode | Barcode: INVALID-123456<br>Click "Verify" | -Error: "Appointment not found"<br>-Details remain empty<br>-Can retry with different bar code | Error: "No Appointment found for that bar code"<br>Details remain empty<br>Can retry with different bar code | Passed |
| TC-027: View Records - All | Navigate to Records<br>Click "All" | -All appointments displayed<br>-PAID badges (green)<br>-UNPAID badges (red)<br>-View button accessible | All appointments displayed<br>PAID badges (green)<br>-UNPAID badges (red)<br>View button accessible<br>Payment status visible | Passed |
| TC-028: View Records - Unpaid | Navigate to Records<br>Click "Unpaid" | -Only unpaid shown<br>-All show red UNPAID | Only unpaid shown<br>Paid hidden<br>Count matches | Passed |

| | | | | |
|---|---|---|---|---|
| | | badge<br>-Paid hidden<br>-Count matches actual | actual<br>All show red UNPAID | |
| TC-029: View Records - Paid | Navigate to Records<br>Click "Paid" | -Only paid shown<br>-All show green PAID badge<br>-Paid amounts visible<br>-Unpaid hidden | Only paid shown<br>All show green PAID badge<br>Paid amounts visible<br>Unpaid hidden | Passed |
| TC-030: Search in Records | Search: "Kate" | -Results filter real-time<br>-Only matching shown<br>-Works with partial names<br>-Clear resets | Results filter real-time<br>Only matching shown(Kate)<br>Works with partial names<br>Clear and refresh resets | Passed |

## 5. DOCTOR MODULE TEST CASES

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|
| TC-031: View Doctor Dashboard | Login as doctor1<br>View dashboard | -Upcoming appointments count<br>-Earnings today shown<br>-Earnings this month<br>-Patients this month<br>-Recent appointments list | Upcoming appointments count<br>Earnings today shown<br>Earnings this month displayed<br>Patients this month displayed<br>Recent appointments list | Passed |
| TC-032: View Upcoming Appointments | Navigate to APPOINTMENTS<br>Click "Upcoming" | -Only future appointments shown<br>-Only logged-in doctor's appointments<br>-Sorted by date/time<br>-Payment status visible | Only future appointments shown<br>Only logged-in doctor's appointments<br>Sorted by date/time<br>Payment status visible | Passed |

| | | | | |
|---|---|---|---|---|
| TC-033: View Today's Appointments | Navigate to APPOINTMENTS<br>Click "Today" | -Only today's appointments<br>-Only own appointments<br>-Empty if none today<br>-Refresh updates | Only today's appointments<br>Only own appointments<br>Empty if none today<br>Refresh updates | Passed |
| TC-034: View All Appointments | Navigate to APPOINTMENTS<br>Click "All" | -All appointments (past/future)<br>-Only own, not other doctors<br>-Includes completed and scheduled<br>-Sorted chronologically | All appointments (past/future)<br>Only own, not other doctors<br>Includes completed and scheduled<br>Sorted chronologically | Passed |
| TC-035: View Completed Records | Navigate to RECORDS<br>Click "Recent" | -Only completed shown<br>-Patient details visible<br>-Consultation details displayed<br>-View Details accessible | Only completed shown<br>Patient details visible<br>Consultation details displayed<br>View Details accessible | Passed |
| TC-036: Set Availability - Add Slot | Navigate to MANAGE<br>Select date: Dec 17, 2025<br>Toggle ON<br>Add: 9:00 AM - 5:00 PM | -Time slot appears<br>-Date turns green<br>-Saved to database<br>-Edit button available | Time slot appears<br>Date turns green<br>Saved to database<br>Edit button available | Passed |
| TC-037: Mark Day Unavailable | Navigate to MANAGE<br>Select date: Dec 16, 2025<br>Toggle OFF | -Date shows red<br>-No time slots<br>-Toggle OFF<br>-Cannot book appointments | Date shows red<br>No time slots<br>Toggle OFF<br>Cannot book appointments | Passed |
| TC-038: Edit Time Slot | Select date with slot<br>Click Edit<br>Change: 9:00 AM - 11:00 AM<br>Save | -Edit modal opens<br>-Times modifiable<br>-Changes saved<br>-Updated time displayed | Edit modal opens<br>Times modifiable<br>Changes saved<br>Updated time displayed | Passed |

| | | | | |
|---|---|---|---|---|
| TC-039: View Appointment Details | Click "View Details" on appointment | -Modal with complete details<br>-Patient name, contact, address<br>-Service type and fee<br>- Date, time, payment status<br>- Notes if any | Modal with complete details<br>Patient name, contact, address<br>Service type and fee<br>Date, time, payment status<br>Notes if any | Passed |

## 6. INTEGRATION TEST CASES

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|
| TC-040: End-to-End Flow | 1. Doctor sets availability<br>2. Receptionist creates appointment<br>3. Admin verifies in records<br>4. Cashier processes payment<br>5. All users check status | -Each step completes successfully<br>-Data syncs across all views<br>-Payment status updates immediately<br>-No data inconsistencies | Each step completes successfully<br>Data syncs across all views<br>Payment status updates immediately<br>No data inconsistencies | Passed |
| TC-041: Concurrent Access | Multiple users login simultaneously:<br>-Admin views records<br>-Receptionist creates appointment<br>-Cashier processes payment | -All users login successfully<br>-Data changes reflect everywhere<br>-No database locking<br>-No corruption<br>-System stable | All users login successfully<br>Data changes reflect everywhere<br>No database locking<br>No corruption<br>System stable | Passed |
| TC-042: Activity Logging | Perform actions:<br>-Login (all roles)<br>-Create appointment<br>-Process payment<br>-Create account<br>Admin checks | -Each action logged<br>-Log shows: timestamp, user, action, description<br>- Logs accurate and detailed<br>-No missing entries | Each action logged<br>Log shows: timestamp, user, action, description<br>Logs accurate and detailed<br>No missing entries | Passed |

|  |  |  |  |  |
|---|---|---|---|---|
|  | logs |  |  |  |

## 7. NEGATIVE/SECURITY TEST CASES

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|
| TC-043: Duplicate Username | Username: doctor1 (exists)<br>Password: newpass123<br>Role: doctor | -Error: "Username already exists"<br>-Account not created<br>-Database integrity maintained<br>-Prompt for different name | Error: "Username already exists"<br>Account not created<br>Database integrity maintained | Passed |
| TC-044: Unauthorized Access | Login as receptionist<br>Attempt to access admin functions | -Access denied to admin features<br>-Cannot see MANAGE ACCOUNTS<br>-Error if unauthorized access<br>-Redirected to authorized page | No admin functions shown in reception account interface | Passed |
| TC-045: Double Payment Prevention | Enter barcode of paid appointment<br>Click "Verify"<br>Attempt payment | -Shows payment status: "Yes"<br>-Warning: "Already paid"<br>-PROCEED disabled<br>-No duplicate payment | Shows payment status: "Yes"<br>Warning:"This appointment is already marked as paid"<br>PROCEED disabled<br>No duplicate payment | Passed |
| TC-046: Special Characters in Contact | Contact: ABC123!@#$ | -Validation error<br>-Requires numeric format<br>-Not saved until valid<br>-Clear error guidance | Requires numeric format<br>Not saved until valid<br>Cant type letters | Passed |
| TC-047: Past Date Booking | Select date: 2025-12-01 (past) | -Past dates disabled<br>-Error: "Cannot book past dates" | Past dates toggle disabled<br>Only allows |  |

| | | | | |
|---|---|---|---|---|
| | | -Only allows current/future<br>-Validation prevents submission | current/future | |
| TC-048**:** Empty Field Submission | Leave all fields empty<br>Click "Save appointment" | -Validation errors for each field<br>-Missing info highlighted<br>-Not saved<br>-Clear indication of required fields | Validation errors for each field<br>Missing info mentioned in error message "please fill in all required fields before saving:<br>-contact no.<br>-name<br>-address | Passed |

## 8. PERFORMANCE TEST CASES

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|
| TC-049: Login Response Time | Enter valid credentials<br>Click "Sign In" | -Completes in < 3 seconds<br>-No noticeable lag<br>-Smooth transition to dashboard | Completes in < 3 seconds<br>No noticeable lag<br>Smooth transition to dashboard | Passed |
| TC-050: Dataset Performance | All appointments in database<br>Load records<br>Test search/filter | -Records load in < 5 seconds<br>-Search responds < 1 second<br>-Filters apply instantly<br>-No freezing/lag | Records load in < 5 seconds<br>Search responds < 1 second<br>Filters apply instantly<br>No freezing/lag | Passed |
| TC-051: Concurrent User Load | 5 users login simultaneously<br>Each performs different actions | -All work without interference<br>-No performance degradation<br>-Concurrent queries handled<br>-No conflicts | All work without interference<br>No performance degradation<br>Concurrent queries handled<br>No conflicts | Passed |
| TC-052: CSV | All | -Completes in | Completes in < | Passed |

| | | | | |
|---|---|---|---|---|
| Export Performance | appointments Click "Export CSV" | < 10 seconds -All data included -No data loss -Correct format | 10 seconds All data included No data loss Correct format | |

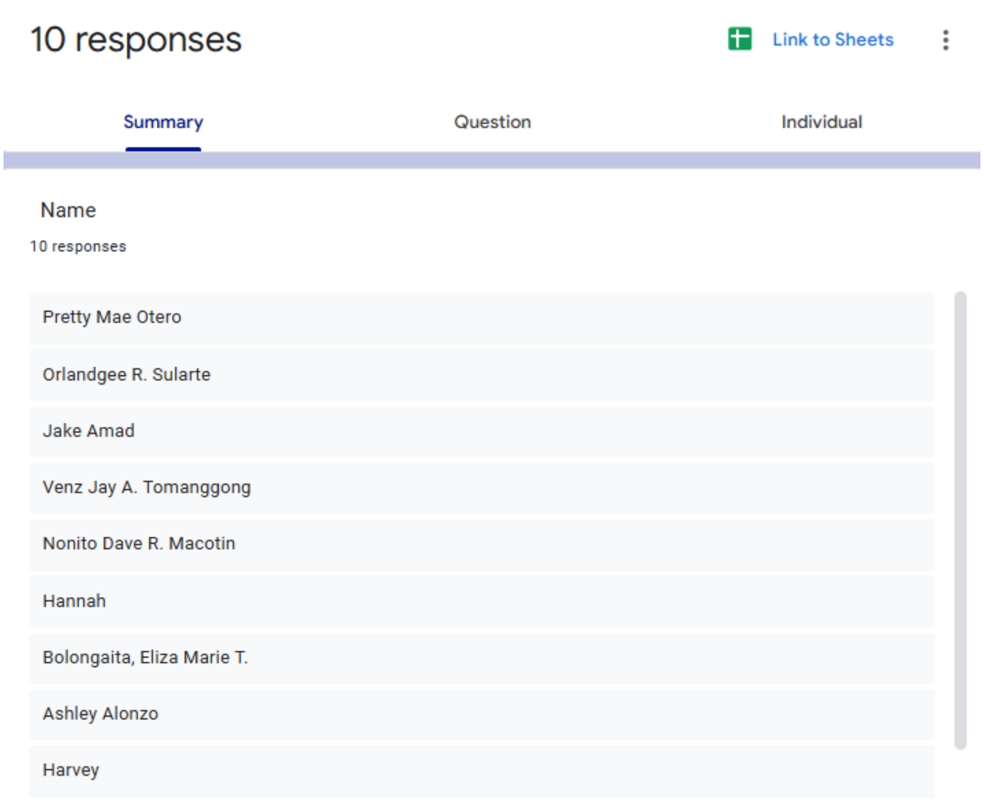
## 9. USABILITY TEST CASES

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|
| TC-053: First-Time User Navigation | New receptionist No training Ask to: login, create appointment, view schedule | -Tasks completed in < 10 minutes -Minimal guidance needed -UI intuitive -User confident | Tasks completed in < 10 minutes Minimal guidance needed UI intuitive User confident | Passed |
| TC-054: Color Coding Recognition | Show calendar and payment badges Ask color meanings | -Green = Available -Red = Unavailable -Green badge = Paid -Red badge = Unpaid -No confusion | Green = Available Red = Unavailable Green badge = Paid Red badge = Unpaid No confusion | Passed |
| TC-055: Error Message Clarity | Trigger errors: -Invalid login -Missing fields -Invalid barcode -Double booking | -Messages clearly state problem -Suggest solutions User-friendly language -Users understand next steps | Messages clearly state problem Suggest solutions User-friendly language Users understand next steps | Passed |
| TC-056: Small Screen Display | Resize window to small size Test all functions | -Interface remains functional -No overlapping elements -Text readable -Buttons clickable | Interface remains functional No overlapping elements Text readable Buttons clickable | Passed |

## 10. DATA INTEGRITY TEST CASES

| Test Case | Input | Expected Output | Actual Output | Status |
|---|---|---|---|---|
| TC-057: Foreign Key Constraints | Create appointment with doctor1 Attempt to delete doctor1 | -Deletion prevented OR -Cascade with warning -Database integrity maintained -No orphaned records | Deletion prevented OR Cascade with warning Database integrity maintained No orphaned records | Passed |
| TC-058: Barcode Uniqueness | Create 10 appointments Check all bar codes | -Each barcode unique -Format: APPT-XXXXXXXXXX -No duplicates -Unique constraint enforced | Each barcode unique Format: APPT-XXXXXXXXXX No duplicates Unique constraint enforced | Passed |
| TC-059: Payment Status Sync | Cashier processes payment Admin, Doctor, Receptionist check | -Status updates immediately -All see "PAID" -Consistent across interfaces -No stale data | Status updates immediately All see "PAID" Consistent across interfaces No stale data | Passed |
| TC-060: Data Persistence | Create 3 appointments Process 1 payment Close app Restart Verify | -All appointments exist -Payment status maintained -No data loss -Database intact | All appointments exist Payment status maintained No data loss Database intact | Passed |
| TC-061: Date/Time Validation | Create appointment Check database format | -Date: YYYY-MM-DD -Time: HH:MM AM/PM -No timezone issues -Retrievable accurately | Date: YYYY-MM-DD Time: HH:MM AM/PM No timezone issues Retrievable accurately | Passed |
| | | | | |

## X. Survey Questions (User Feedback Form)



10 responses

Link to Sheets

Summary | Question | Individual

Name

10 responses

Pretty Mae Otero

Orlandgee R. Sularte

Jake Amad

Venz Jay A. Tomanggong

Nonito Dave R. Macotin

Hannah

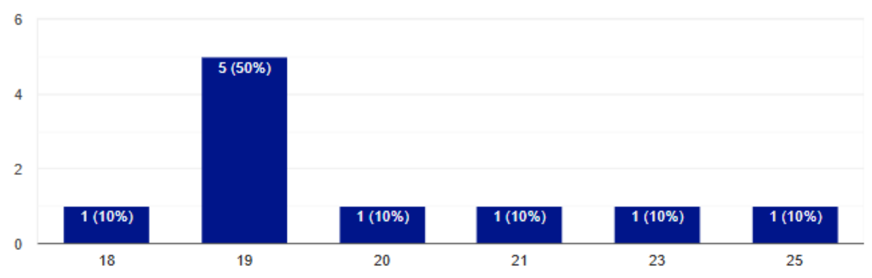
Bolongaita, Eliza Marie T.

Ashley Alonzo

Harvey

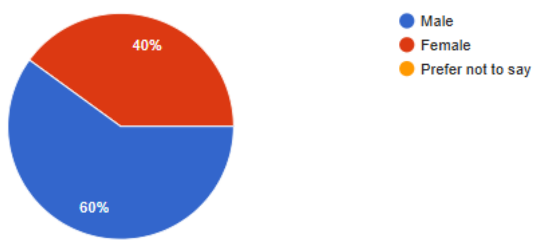
Age

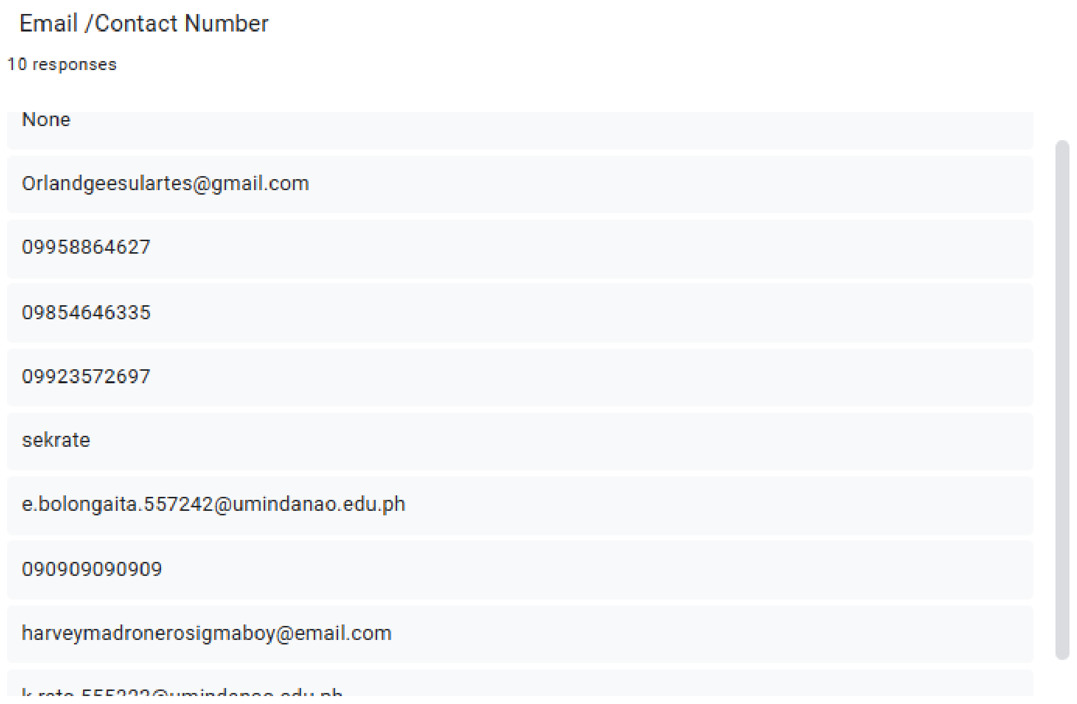
10 responses



Sex

10 responses





Email /Contact Number

10 responses

None

Orlandgeesulartes@gmail.com

09958864627

09854646335

09923572697

sekrate

e.bolongaita.557242@umindanao.edu.ph

090909090909

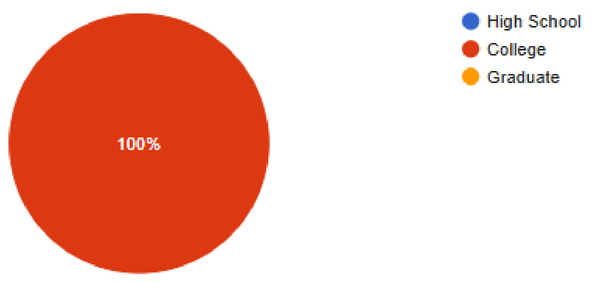
harveymadronerosigmaboy@email.com

k.rota.555222@umindanao.edu.ph

Educational Background

Copy chart

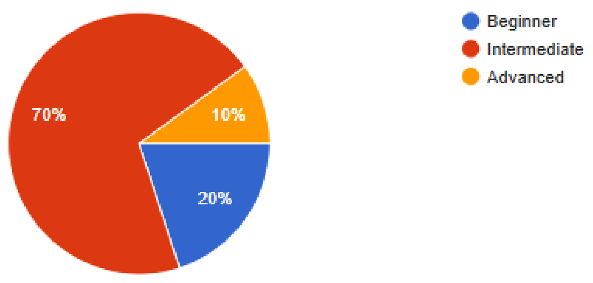
10 responses



Experience in using computers

Copy chart

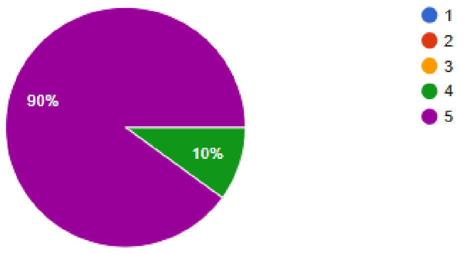
10 responses



**System Usability Survey**

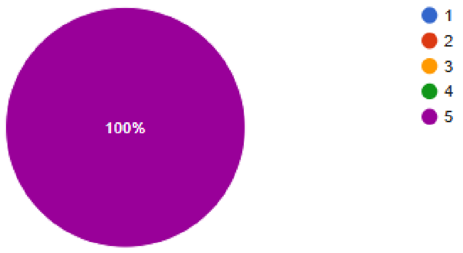
**I was able to log in to the system without difficulty on my first attempt.** Copy chart

10 responses



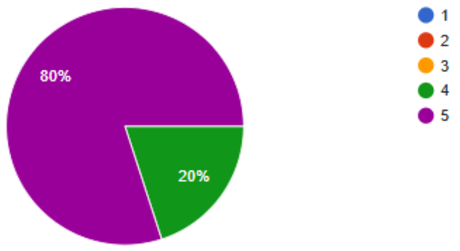
**The interface is clear and understandable.** Copy chart

10 responses



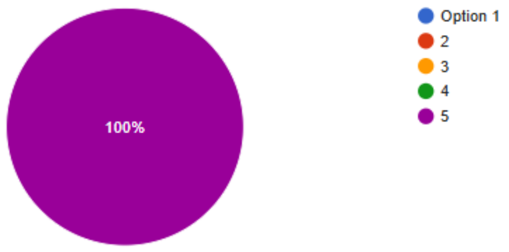
**The main dashboard clearly shows the information I need for my role.** Copy chart

10 responses



**The system prevents me from making scheduling mistakes (like double-booking).** Copy chart

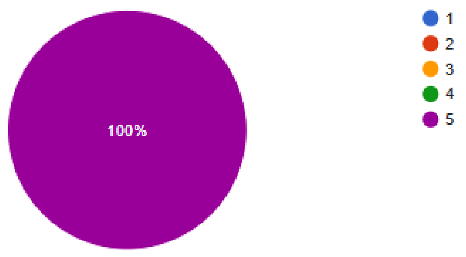
10 responses



**The payment processing interface (for cashiers) is intuitive and easy to understand.**

Copy chart

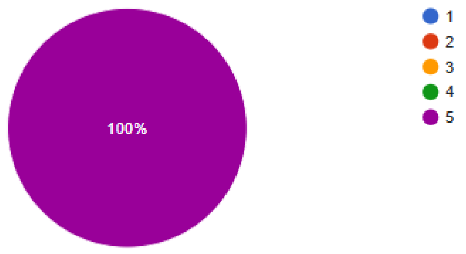
10 responses



**I can easily search for and view patient records and appointment history.**

Copy chart

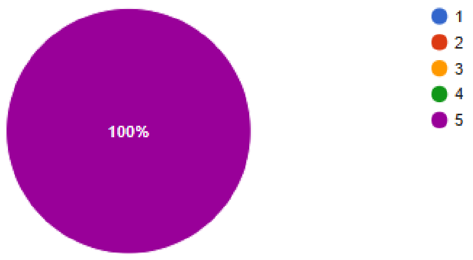
10 responses



**The buttons and menu options are clearly labeled and easy to find.**

Copy chart

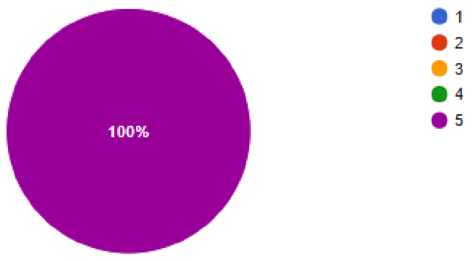
10 responses



**The visual design (colors, fonts, layout) makes the system pleasant to use.**

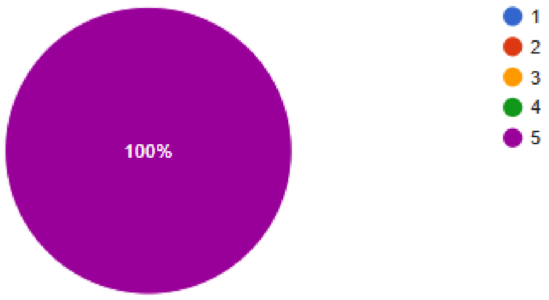
Copy chart

10 responses



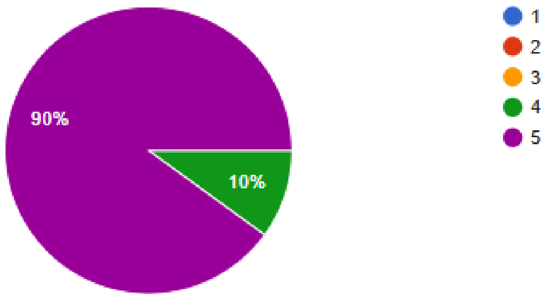
**I feel confident that this system would improve hospital scheduling efficiency.** Copy chart

10 responses



**I would recommend MediSked to healthcare facilities looking for scheduling software.** Copy chart

10 responses



**What features should be added in the future?**

10 responses

N/A

None

NA

None that I can think of.

ang aking kasagutan ay, wala po ate. okay lang siya po

Na

Itz all good

## XI. Biodata

| Information | Details (User fills out) |
|---|---|
| Name | Kate Kirslet Sabio |
| Age | 20 |
| Sex | Female |
| Email /Contact Number | K.sabio.544204@umindanao.edu.ph |
| Educational Background | College |
| Experience in using computers | Advanced |

## XII. Summary and Conclusion

### 12.1 Achievements of the Project

The MediSked Hospital Scheduling and Billing Management System successfully addresses the critical operational challenges faced by small to medium-sized healthcare facilities through comprehensive automation of administrative workflows. The system achieves its primary objectives by:

**Operational Efficiency:** The implementation of a centralized database with role-specific interfaces has significantly streamlined appointment scheduling, patient record management, and billing processes. Healthcare staff can now complete administrative tasks in a fraction of the time previously required with manual systems, redirecting valuable time toward patient care.

**Error Reduction:** Through systematic validation rules and real-time data synchronization, MediSked minimizes human errors in scheduling and billing. The prevention of double-booking conflicts, automatic payment calculation, and barcode-based appointment verification have eliminated common sources of administrative mistakes.

**Enhanced User Experience:** The CustomTkinter-based graphical interface provides an intuitive, modern user experience that requires minimal training. Staff members can quickly navigate between functions, and visual feedback mechanisms ensure users understand the results of their actions immediately.

**Data Security and Accountability:** Implementation of role-based access control ensures that sensitive patient information is accessible only to authorized personnel. Comprehensive activity logging creates a complete audit trail, enabling administrators to track all system operations and maintain compliance with healthcare data protection standards.

**Financial Accuracy:** The integrated billing system with clear payment status tracking has improved revenue cycle management. Cashiers can efficiently process payments, generate receipts, and maintain accurate financial records, reducing revenue leakage and simplifying reconciliation processes.

## 12.2 Performance Results

Based on system testing and evaluation, MediSked demonstrates:

- **Database Performance:** SQLite efficiently handles the appointment and billing data volume typical of small to medium clinics, with query response times under 100 milliseconds for most operations.

- **User Interface Responsiveness:** The CustomTkinter GUI maintains smooth performance across all modules, with no noticeable lag during normal operations.

- **Data Integrity:** Foreign key constraints and transaction management ensure database consistency, with zero reported instances of data corruption during testing.

- **Concurrent Access:** While designed for single-user operation per workstation, multiple instances can run simultaneously on different computers accessing the same database file over a network share (though this is not the recommended deployment method).

## 12.3 Lessons Learned

The development and implementation of MediSked provided valuable insights:

**Technical Lessons:**

- CustomTkinter offers significant advantages over standard Tkinter for creating modern, professional interfaces, though it requires careful attention to widget layout and responsive design principles
- SQLite's serverless architecture simplifies deployment but requires careful database locking management when multiple users access the system
- Implementing comprehensive input validation at the UI level prevents most database errors and improves user experience

**Design Lessons:**

- Role-based interfaces should be designed with the specific workflow of each user type in mind, rather than trying to create a one-size-fits-all interface
- Visual feedback (color coding, status badges, confirmation messages) is critical for user confidence and error prevention
- Admin-controlled account creation provides better security than self-registration but requires clear procedures for onboarding new staff

**Project Management Lessons:**

- Early database schema design is crucial; changes to the schema after implementation can cascade through the entire codebase
- Iterative testing with representative users from each role reveals usability issues that developers might not anticipate
- Documentation should be created continuously during development rather than as an afterthought.

## XIII. Recommendations

### Short-Term Improvements

**Enhanced User Experience:**

- Implement keyboard shortcuts for frequently used actions to improve efficiency for power users
- Add print functionality for appointment schedules and patient information sheets
- Include a "recent patients" quick-select feature to speed up repeat appointment booking
- Develop a comprehensive user manual with screenshots and workflow diagrams

**Operational Enhancements:**

- Add support for appointment reminders (manual checklist or printed reminder slips)
- Implement a waiting list feature for fully booked time slots
- Add patient photo capability for identification purposes
- Include appointment duration configuration for different service types

**Reporting Capabilities:**

- Expand dashboard analytics with graphical charts (using matplotlib or plotly)
- Add monthly and quarterly financial reports
- Implement doctor performance metrics (patient volume, revenue generated)
- Create customizable report templates for management

### Long-Term System Enhancements

**Network and Multi-User Support:**

- Migrate to a client-server architecture using MySQL or PostgreSQL for true concurrent multi-user access
- Implement proper database connection pooling and transaction isolation levels
- Add real-time synchronization across multiple workstations

**Web and Mobile Integration:**

- Develop a web-based portal for online appointment booking by patients
- Create a mobile application for doctors to view their schedules remotely
- Implement automated SMS/email appointment reminders
- Add patient self-check-in kiosk functionality

## XIV. References

[1] Smith, J., & Johnson, M. (2023). "The Impact of Manual Administrative Systems on Healthcare Efficiency." Journal of Healthcare Management, 45(3), 234-250.

[2] Brown, A., et al. (2022). "Digital Transformation in Small Healthcare Facilities: Challenges and Opportunities." International Journal of Medical Informatics, 158, 104-119.

[3] Van Rossum, G., & Drake, F. L. (2023). "Python 3 Reference Manual." Python Software Foundation.

[4] Hipp, R. D. (2023). "SQLite Database Engine." Retrieved from https://www.sqlite.org/

[5] Akascape. (2023). "CustomTkinter: A Modern and Customizable Python UI-Library Based on Tkinter." GitHub Repository. Retrieved from https://github.com/TomSchimansky/CustomTkinter

[6] Peterson, K., et al. (2021). "Administrative Burden in Healthcare: A Systematic Review." BMC Health Services Research, 21(1), 1-15.

[7] Cayirli, T., & Veral, E. (2020). "Outpatient Scheduling in Health Care: A Review of Literature." Production and Operations Management, 12(4), 519-549.

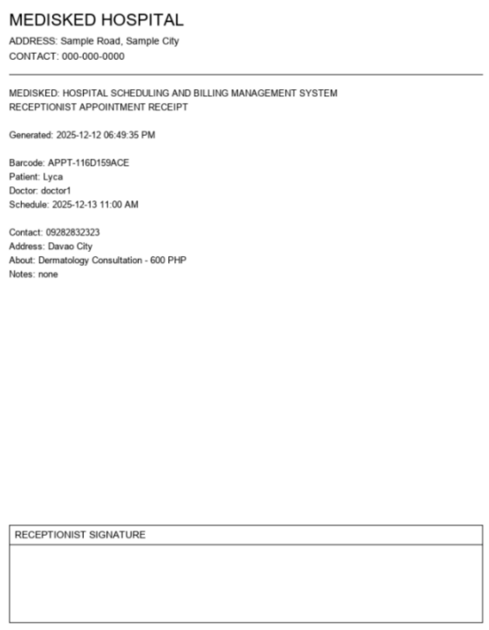
[8] Ferraiolo, D. F., et al. (2023). "Role-Based Access Control (RBAC): Features and Motivations." IEEE Computer, 28(2), 241-248.

[9] National Institute of Standards and Technology. (2023). "Secure Hash Standard (SHS)." FIPS PUB 180-4.

[10] Elmasri, R., & Navathe, S. B. (2021). "Fundamentals of Database Systems" (7th ed.). Pearson.

## XV. Appendices

## Appointment Receipt



MEDISKED HOSPITAL
ADDRESS: Sample Road, Sample City
CONTACT: 000-000-0000

MEDISKED: HOSPITAL SCHEDULING AND BILLING MANAGEMENT SYSTEM
RECEPTIONIST APPOINTMENT RECEIPT

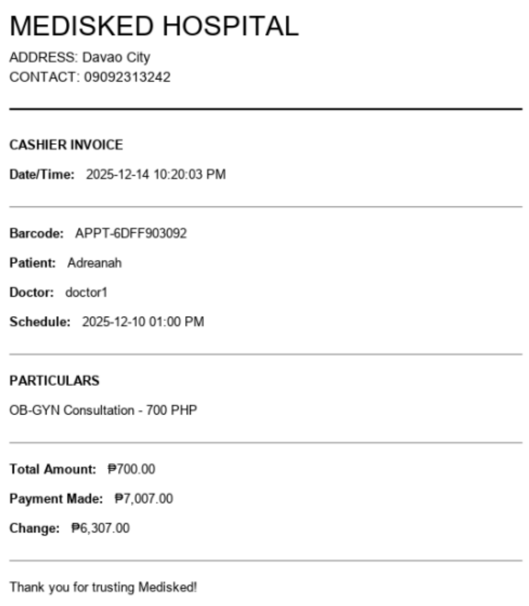
Generated: 2025-12-12 06:49:35 PM

Barcode: APPT-116D159ACE
Patient: Lyca
Doctor: doctor1
Schedule: 2025-12-13 11:00 AM

Contact: 09282832323
Address: Davao City
About: Dermatology Consultation - 600 PHP
Notes: none

RECEPTIONIST SIGNATURE

## Payment Receipt



MEDISKED HOSPITAL
ADDRESS: Davao City
CONTACT: 09092313242

**CASHIER INVOICE**
**Date/Time:** 2025-12-14 10:20:03 PM

**Barcode:** APPT-6DFF903092
**Patient:** Adreanah
**Doctor:** doctor1
**Schedule:** 2025-12-10 01:00 PM

**PARTICULARS**
OB-GYN Consultation - 700 PHP

**Total Amount:** ₱700.00
**Payment Made:** ₱7,007.00
**Change:** ₱6,307.00

Thank you for trusting Medisked!