Secure Hospital Management System (Kivy/Python)

This document provides an overview of the Hospital Management System (HMS), a standalone desktop application built using the Kivy framework in Python. The application is designed with a strong focus on data security, including encryption and strict auditing.

1. Architecture Overview

The application is fully self-contained within the `main.py` file, incorporating both the front-end (Kivy UI) and back-end (Python logic, SQLite database management, and encryption).

Key Architectural Components:

Component	Technology/Concept	Purpose
User Interface	Kivy (Python UI Framework)	Provides a responsive, cross-platform graphical interface for users.
Data Security	Fernet (AES-128 Symmetric Encryption)	Encrypts sensitive patient data *at rest* within the database.
Persistent Storage	SQLite (Embedded Database)	Uses two separate database files for data isolation.
Audit Logging	Dedicated Audit Database	Immutable record of all key user and system actions (logins, deletions, etc.).
Security Assurance	Parameterized Queries	Prevents SQL Injection attacks in database operations.

2. Security Features

A. Data Isolation and Integrity

The application utilizes a **Dual Database Architecture** for security and compliance:

1. **`hms_patient_data.db`**: Stores sensitive patient records. All fields containing personally identifiable information (Name, Condition) are **encrypted** using Fernet/AES.
2. **`hms_audit_log.db`**: Stores non-repudiable logs of all critical actions (logins, patient adds/deletes). This database is kept separate to maintain log integrity, even if the primary data store were compromised.

B. Encryption (Fernet/AES)

A unique, locally stored key (`hms_key.key`) is used by the Fernet library to encrypt and decrypt sensitive data. The data is only decrypted momentarily when displayed on the Dashboard's **Patient Records** screen.

C. SQL Injection Prevention

The **SQLI Assurance Test** module demonstrates that all database queries use **safe parameterization**. This means user input is treated as literal values, not executable code, making common injection techniques (e.g., `' OR '1'='1 --`) ineffective.

3. Getting Started

A. Prerequisites

- Python 3.x
- Kivy framework
- `cryptography` library (for Fernet encryption)

B. Installation

Before running, ensure necessary libraries are installed:

```
pip install kivy python-for-android
pip install cryptography
```

C. Running the Application

Execute the single file:

```
python main.py
```

D. Default Credentials

The application starts on the **Login Screen**.

Field	Value
Username	`admin`
Password	`password123`

4. Key Modules

1. **Patient Records**: Add new patients (data is encrypted), view the patient list (data is decrypted on retrieval), and securely delete records.
2. **SQLI Assurance Test**: Allows an administrator to test the system's resilience against SQL injection using real database queries.
3. **Audit Log Report**: Displays the time-stamped, user-specific log of all activity, demonstrating compliance and accountability.