Architecture & Design

Tech Stack Utilized

Frontend (Mobile App): Java with Android SDK (XML for layouts)

Database: SQLite (local, relational database)

Notification System: Android AlarmManager + BroadcastReceiver

Design Pattern: MVVM-inspired with direct data handling (owing to limited app scope)

Why This Stack Was Utilized

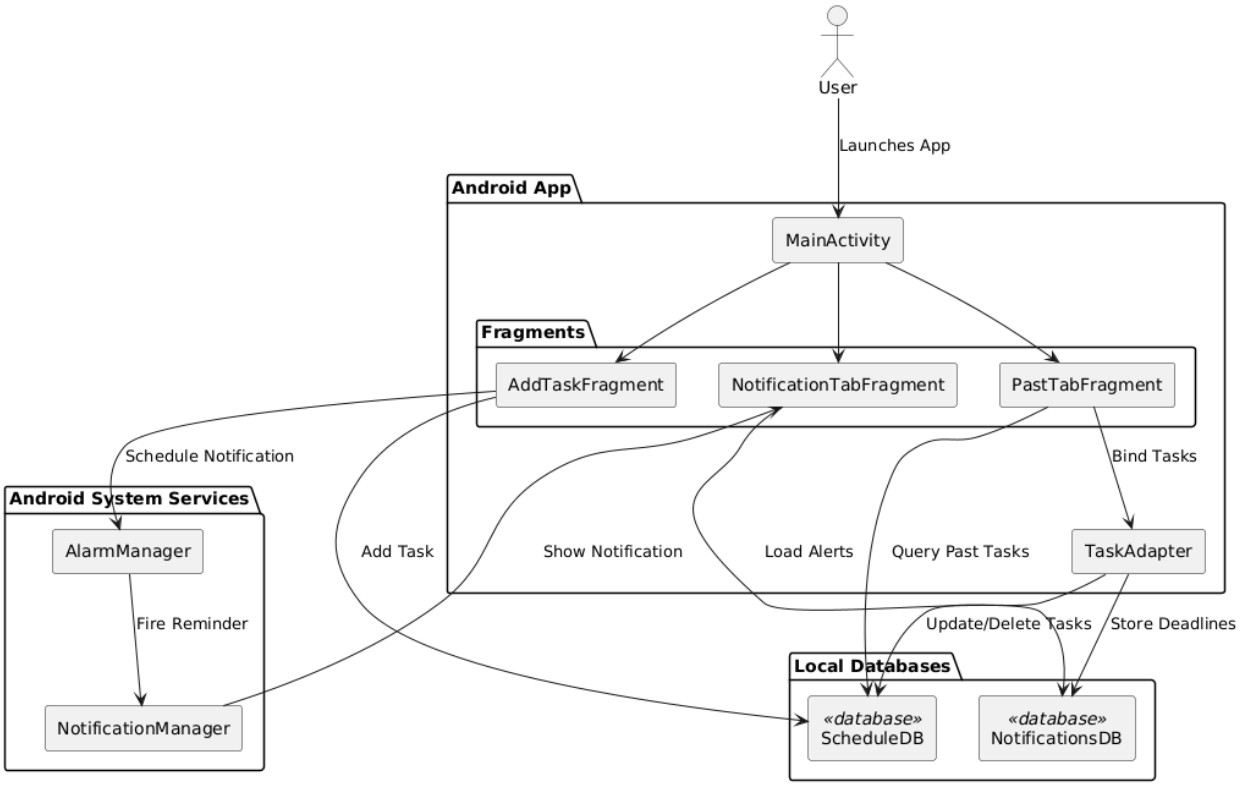
Java & Android SDK: Java is still well-supported in Android development. It's known and gives fine-grained control of views and components without extra abstraction layers like Jetpack Compose.

SQLite: Local storage support built-in, no library dependencies, ideal for offline-first task-oriented applications.

AlarmManager: Scheduled task notification capability without persistent background services (more power-efficient).

Manual MVVM Logic: Simple-mindedly, full MVVM was not strictly implemented (e.g., ViewModel, LiveData), but the separation of code is done to some degree—fragments manage the UI, and database classes manage logic.

Architecture Diagram

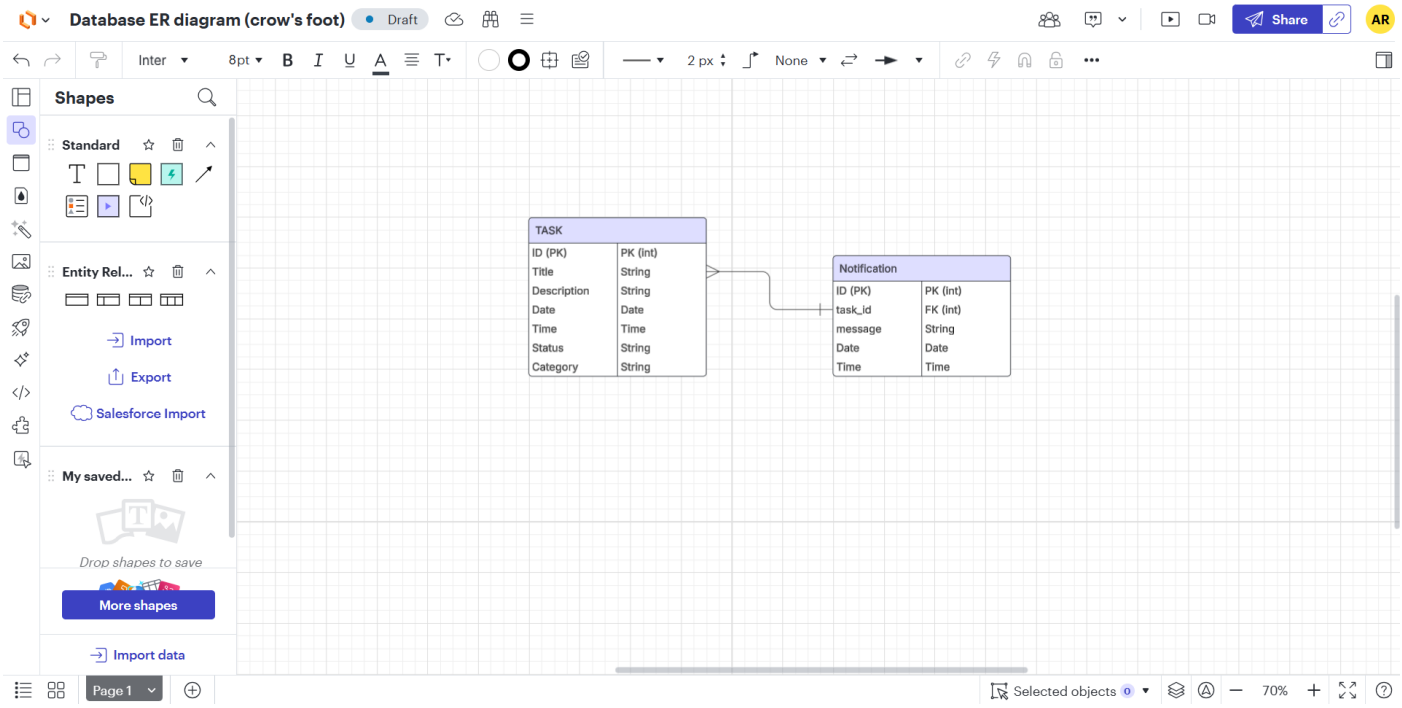


This diagram illustrates the core components and data flow of the Task Scheduler Android application. The app uses **SQLite databases** (ScheduleDB and NotificationsDB) for persistent storage of tasks and notifications. Users interact with the UI through multiple fragments (AddTaskFragment, PastTabFragment, and NotificationTabFragment) managed by the MainActivity.

The AddTaskFragment allows users to create tasks, which are stored in ScheduleDB and optionally scheduled via the **AlarmManager** to trigger system notifications through the **NotificationManager**. The PastTabFragment displays completed or past-due tasks using a RecyclerView backed by a custom TaskAdapter, while the NotificationTabFragment fetches alerts from NotificationsDB.

This modular design separates concerns, ensures reusable components, and leverages Android system services for time-based notifications.

Database Scheme



The ER diagram shows two main entities: Task and Notification. Each Task has fields like title, description, date, time, status, and category. A Task can have multiple Notifications, each linked using a task_id foreign key. This design helps manage reminders for tasks and allows future extensions like recurring tasks or user profiles.