# 1. Problem Statement & Objectives

## Problem Statement

Users often forget important events, deadlines, or appointments due to the lack of a simple, effective event reminder application. The goal is to develop a user-friendly Native Android mobile application that allows users to add events, display a countdown for each event, and receive reminders, helping users stay organized and productive.

## Objectives

**•** Develop a Native Android App using Kotlin.  
**•** Implement an intuitive interface that allows users to:  
 - Add, Edit, Delete events.  
 - View countdowns to upcoming events.  
 - Receive notifications/reminders before the event date.  
**•** Use Room Database for local data storage without backend dependency.  
**•** Follow MVVM (Model-View-ViewModel) architecture to ensure scalability and clean code organization.  
**•** Ensure the app works offline and provides fast performance.  
**•** Provide a simple and user-friendly UI/UX design for easy navigation.

## 1.1 Use Case Diagram & Descriptions

+-----------------------+  
| User |  
+-----------------------+  
 |  
+----------+----------+----------+  
| | |  
+-----------+ +--------------+ +----------------+  
| Add Event| | View Events | | Edit/Delete |  
+-----------+ +--------------+ +----------------+  
 |  
 +------------------+  
 | Set Reminder |  
 +------------------+

Use Case Descriptions:  
- User interacts with the application.  
- The user can Add Event by entering event details.  
- The user can View Events, showing the list of all events with countdown timers.  
- The user can Edit/Delete any existing events.  
- After adding or editing, the system Sets Reminders to notify the user before the event.

## 1.2 Functional & Non-Functional Requirements

### Functional Requirements

- Users can register/login into the app. (Optional if you want local only)  
- Users can add/edit/delete events.  
- Users can view a list of events with countdown timers.  
- Users receive notifications/reminders before events.

### Non-Functional Requirements

- The app works offline without requiring internet access.  
- Uses Room Database to store events locally.  
- MVVM architecture for scalable and maintainable code.  
- Simple and intuitive UI/UX for better user experience.  
- The app should load events quickly and handle multiple events efficiently.

## 1.3 Software Architecture

+----------------+  
| View | <-- Activity/Fragment  
+----------------+  
 |  
 v  
+----------------+  
| ViewModel |  
+----------------+  
 |  
 v  
+----------------+  
| Repository |  
+----------------+  
 |  
 v  
+----------------+  
| Room Database |  
+----------------+