ABSTRACT

The Note Taking and To-Do List App is an Android-based mobile application developed using Java and XML in Android Studio. It is designed to help users organize their daily activities more efficiently by providing features such as note-taking, task management. The application offers a simple and effective way to capture, store, and manage personal and professional tasks.

The app enables users to create, update, and delete notes and to-do items, helping them stay organized and focused. It supports both plain text notes and interactive checklists, allowing for flexible task management. The user interface is clean, intuitive, and designed for smooth navigation across features.

For local data storage and management, the application uses the Room persistence library, which serves as an abstraction layer over SQLite. This ensures robust and efficient handling of structured data with compile-time checks, easy data access patterns, and lifecycle-aware components.

By eliminating the need for manual note-taking and improving task tracking, the Note Taking and To-Do List App enhances everyday organization and scheduling. The project utilizes Java for backend logic, XML for UI design, and Room for reliable and scalable data storage.

TABLE OF CONTENTS

INDEX

INTRODUCTION

1. INTRODUCTION

The Note Taking and To-Do List App is an Android mobile application designed to help users to take notes and organize their daily tasks efficiently. It allows users to create, edit, and delete notes and to-do lists, making it easy to manage personal and professional activities in one place.

The app is developed in Android Studio using Java for logic and XML for the user interface, with Room Database for storing data locally. Room ensures reliable and structured storage, allowing the app to work even without an internet connection.

With its simple design and easy navigation, the application provides a quick and effective way for users to stay organized and improve productivity.

SYSTEM STUDY

2. SYSTEM STUDY

2.1 EXISTING SYSTEM

In the existing scenario, most users rely on separate tools for note-taking and task management. Traditional paper-based methods are prone to loss and damage, while digital alternatives often require internet access, are cluttered with unnecessary features, or come with subscription fees.

Many existing mobile applications are either too basic lacking search and customization features or overly complex, making them difficult for casual users to adopt. This creates a demand for an app that offers the right balance between functionality and simplicity.

2.2 PROPOSED SYSTEM

The proposed Note Taking and To-Do List App addresses these gaps by offering a simple, offline, and efficient solution that integrates note-taking and to-do management in one application.

Key features of the proposed system include:

- ❖ Unified Platform**:** Manage both notes and tasks without switching between apps.
- ❖ Offline Functionality: Store and access data without internet connectivity.
- ❖ Robust Data Management: Use Room Persistence Library for efficient CRUD operations with compile-time query validation.
- ❖ User-Friendly Design: A clean and visually appealing UI with smooth navigation.
- ❖ Customizable Notes: Ability to change note backgrounds to improve visual categorization.
- ❖ Quick Retrieval: Built-in search functionality for finding notes instantly.
- ❖ Data Recovery: Undo delete option via Snackbar to prevent accidental data loss.

2.3 PROBLEM DEFINITION AND PROJECT DESCRIPTION

In today’s fast-paced lifestyle, individuals often struggle to keep track of their tasks, events, and important notes. Traditional manual methods, such as paper notebooks or unstructured digital notes, are prone to misplacement, lack searchability, and offer limited task management capabilities.

While there are many existing note-taking and to-do applications, they often come with unnecessary complexity, require constant internet connectivity, or lack seamless integration between notes and checklists. This creates a gap for a lightweight, offline, and user-friendly mobile solution that can manage both personal and professional tasks effectively.

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 REQUIREMENTS SPECIFICATION

HARDWARE REQUIREMENTS

❖ Processor	: Intel i3 or higher
❖ RAM	: 4 GB (8 GB recommended)
❖ Hard Disk	: 40 GB or more (for IDE, SDKs, and project files)
❖ Monitor	: 15” VGA/LED monitor
❖ Keyboard	: Standard 104 keys keyboard
❖ Mouse	: Optical mouse
❖ Mobile Device for Testing	: Android smartphone, minimum 2 GB RAM, Android 6.0 or higher, 50 MB free storage

SOFTWARE REQUIREMENTS

❖ Operating System	: Windows 10 / 11 (64-bit) or macOS / Linux
❖ IDE	: Android Studio (latest stable version)
❖ Programming Languages	: Java (Backend Logic), XML (UI Design)
❖ Database	: Room Persistence Library
❖ Build System	: Gradle
❖ Android SDK	: API Level 33 or higher (minSdkVersion 23)
❖ Emulator	: For app testing and debugging

3.2 FEASIBILITY STUDY

Before developing the Note Taking and To-Do List App, a feasibility study was conducted to evaluate whether the project is viable and practical in terms of technology, cost, and usability. The study ensures that the system can be implemented successfully with the available resources. The feasibility study examines the following aspects:

A. TECHNICAL FEASIBILITY

The Note Taking and To-Do List App is technically feasible due to the following reasons:

1. Development Tools
    - ❖ Android Studio provides a robust environment for designing, coding, debugging, and testing Android applications.
    - ❖ Java is a widely used, stable, and well-supported programming language for Android development.
    - ❖ XML is used for designing responsive and customizable layouts.
2. Database & Storage
    - ❖ Room Persistence Library serves as an abstraction over SQLite, offering compile-time query checks, lifecycle awareness, and easy data handling.
    - ❖ Local storage eliminates the dependency on internet connectivity for note-taking and task management.
3. Device Compatibility
    - ❖ Compatible with a wide range of Android devices running API 21 (Lollipop) and above.
    - ❖ Designed to work efficiently even on devices with limited hardware resources.
4. Implementation Complexity
    - ❖ The system’s architecture (Entities, DAO, Repository, UI) ensures modularity, making it easier to implement, debug, and maintain.

B. ECONOMIC FEASIBILITY

The app is economically feasible because:

1. Low Development Cost

- No need for paid development tools; Android Studio and Java are free.
- Uses open-source libraries like Room, reducing licensing costs.

2. No Server Costs
   - The app is fully offline, eliminating expenses related to hosting, cloud storage, or API usage.
3. Maintenance Cost
   - Maintenance is minimal since data is stored locally and does not depend on external services.
4. Scalability without Additional Cost
   - New features can be added without major infrastructure changes, keeping future costs low.

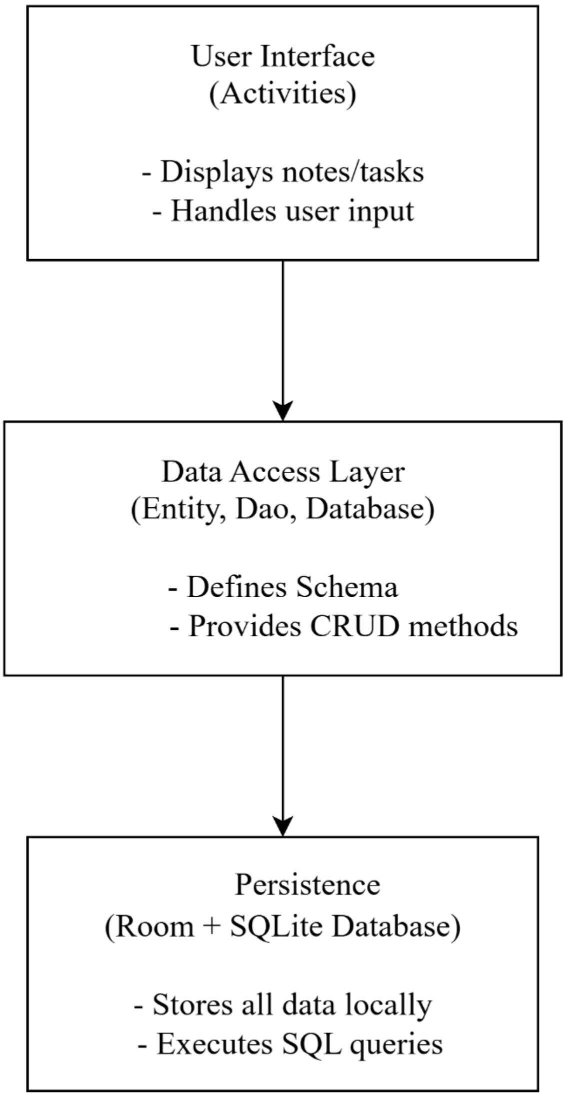
C. OPERATIONAL FEASIBILITY

1. Ease of Use
   - Simple and intuitive user interface designed for all age groups.
   - No steep learning curve — users can start using the app immediately after installation.
2. User Adoption
   - Helps both students and professionals by offering organized note-taking and task management.
   - Enhances productivity by reducing the time spent on manual planning.
3. Maintenance and Support
   - Modular code structure allows easy bug fixes and updates.
   - The app does not require constant supervision or online support.
4. Reliability in Day-to-Day Use
   - Offline functionality ensures that the app is accessible anytime, even without internet.
   - Data persistence through Room ensures that notes remain safe across app restarts and device reboots.

SYSTEM DESIGN

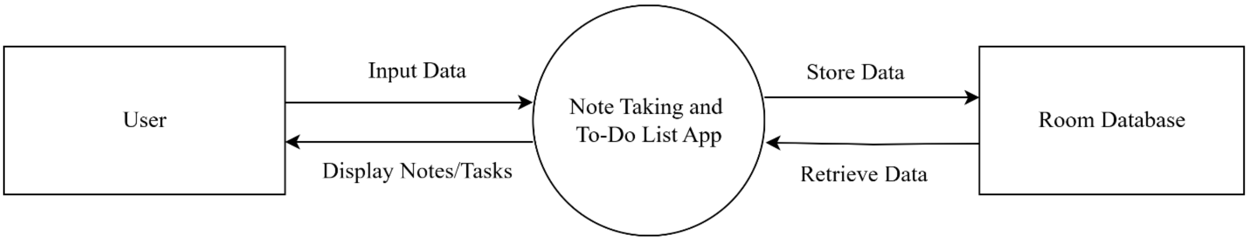
4. SYSTEM DESIGN

4.1 ARCHITECTURAL DESIGN

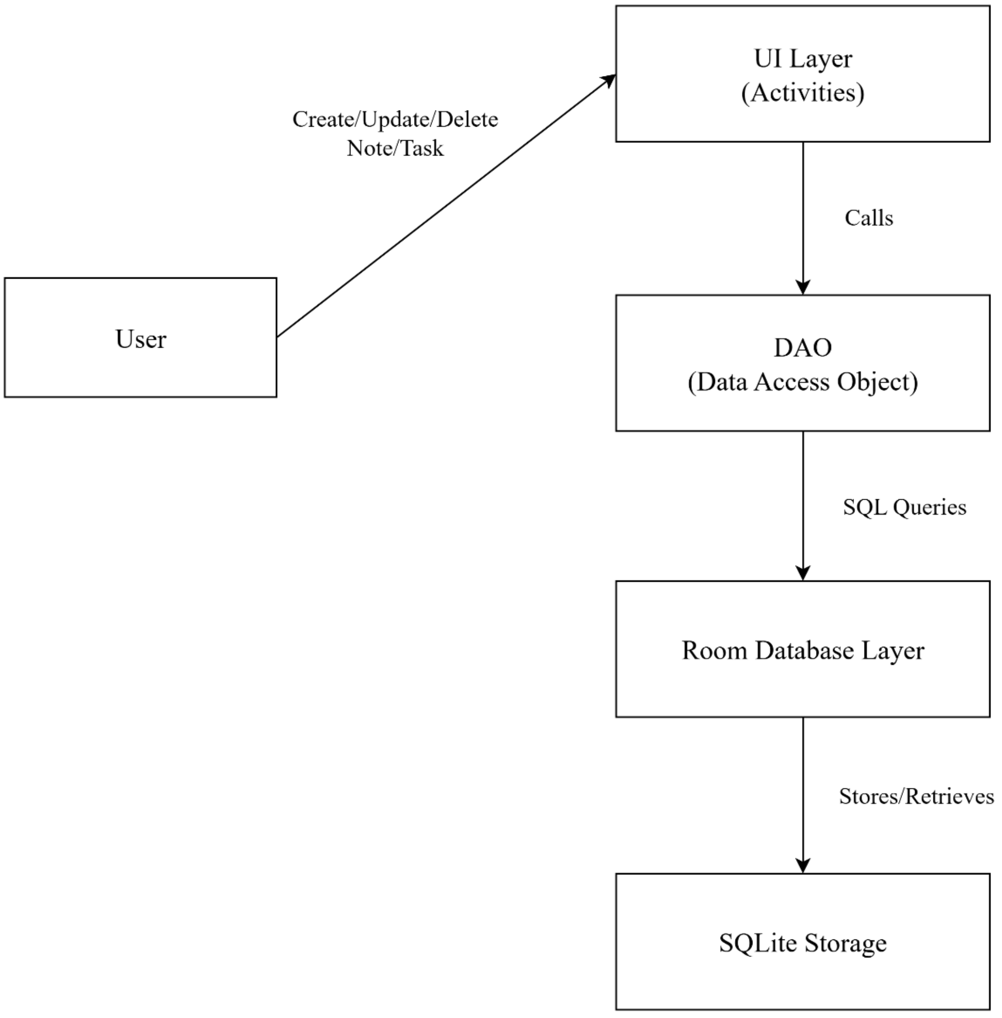


4.2 DATA FLOW DIAGRAM

DFD Level 0 (Context Diagram)

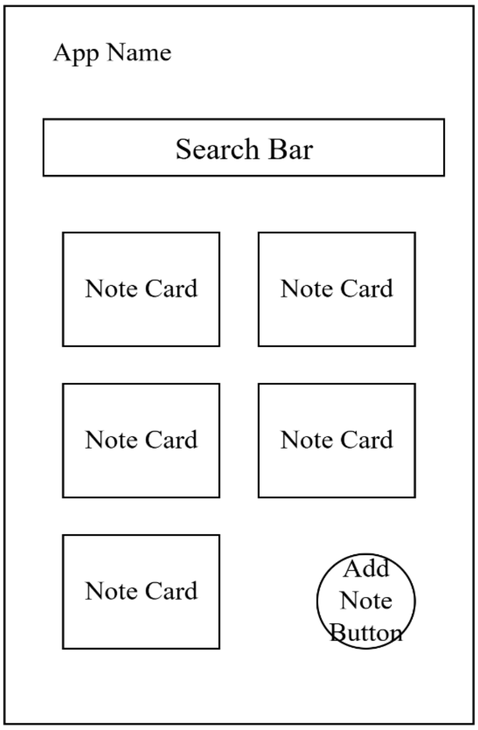


DFD Level 1

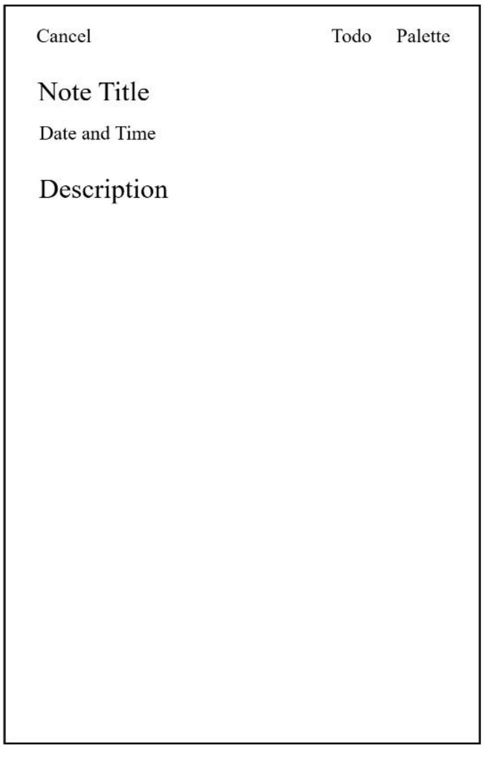


4.3 USER INTERFACE DESIGN

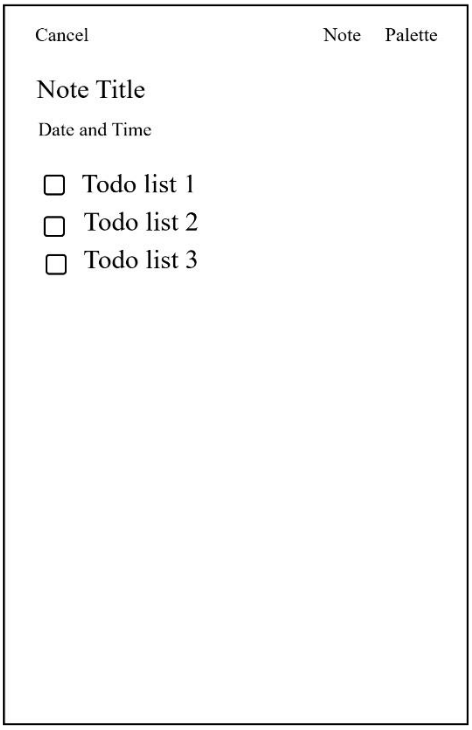
Home Page



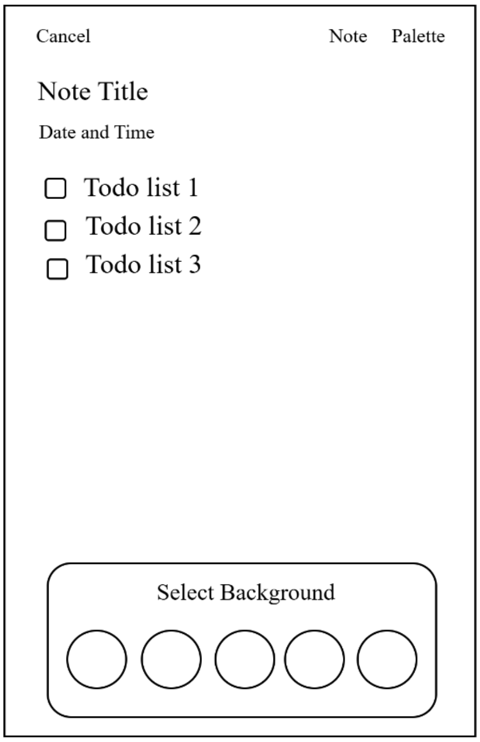
Note Creation Page



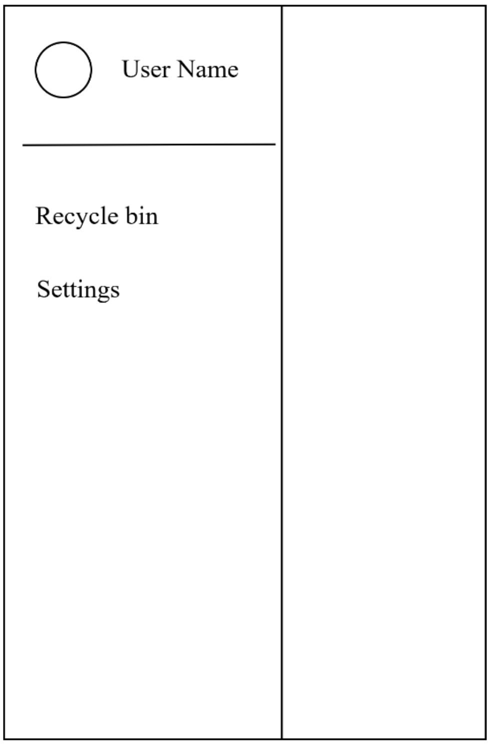
Todo List Mode



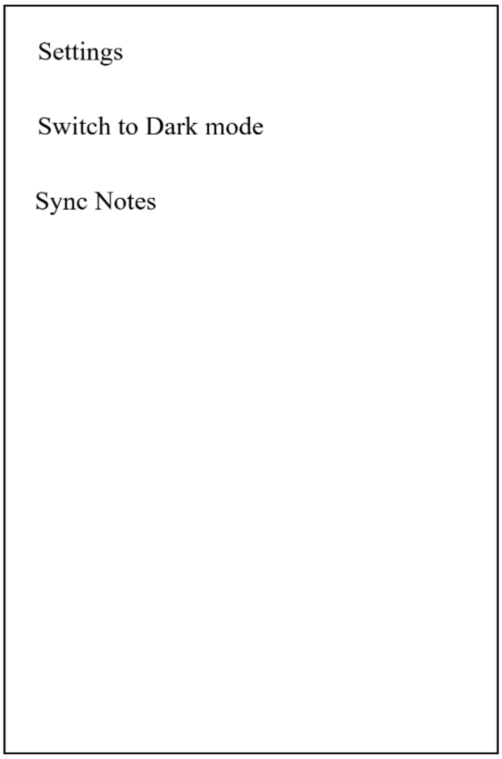
Select Background Color



Side Bar



App Settings



4.4 NORMALIZATION

Normalization is the process of structuring a database to reduce data redundancy and improve data integrity. The notes table in the Note Taking and To-Do List App follows the principles of normalization, ensuring efficient storage and retrieval of information.

First Normal Form (1NF):

- Each column contains only atomic values (e.g., title, date, note, color).
- There are no repeating groups or arrays within the table.
- The todoListJson field stores structured checklist data as a serialized string, maintaining atomicity.

Second Normal Form (2NF):

- The table has a primary key (id) that uniquely identifies each record.
- All non-key attributes (title, date, note, color, etc.) are fully dependent on the primary key.

Third Normal Form (3NF):

- There are no transitive dependencies all attributes depend solely on the primary key.
- Attributes like color or isTodo do not depend on other non-key attributes but only on id.

The database design of the Note Taking and To-Do List App adheres to the Third Normal Form (3NF), ensuring minimal redundancy, consistency in data storage, and ease of maintenance.