Detailed Requirements for Android POS System

1. Core Features

a. Barcode Scanning

- Use the mobile camera for barcode scanning (via libraries like Google ML Kit or ZXing).

- Decode barcodes to identify products in the stock database.

- Display product details (Product Name, MRP, Sale Price, Quantity).

b. Importing Products

- Allow importing of an Excel file in the required format:

  Columns: Barcode, Product Name, MRP, Sale Price, Purchase Price, Quantity.

- Validate the file format before importing.

- Append or update stock based on the imported file.

c. Sales and Inventory Management

- Add products to the cart using the barcode scanner or manual search.

- Update stock quantity dynamically when a sale is made.

- Track and record each sale for reporting and analytics.

d. Invoice Generation

- Generate digital invoices after each transaction.

- Show itemized details: Product Name, Sale Price, Quantity, Total Price.

- Save invoices locally (PDF format).

2. Analytics Dashboard

a. Sales Overview

- Display today's total sales value.

- Show total sales quantity.

b. Profit Analysis

- Calculate profit using:

  Profit = (Sale Price - Purchase Price) x Quantity Sold.

- Show profit for the day, week, and month.

c. Product Performance

- Identify top-selling products.

- Highlight products with low stock.

d. Stock Insights

- Show remaining stock by category or product.

- Alert for low-stock items (display in the dashboard only).

4. User Interface (UI/UX) Requirements

- Simple and Clean Design: Ensure the app is intuitive and easy to use.

- Responsive UI: Smooth performance even on low-end devices.

- Dark Mode: Add support for dark and light themes.

- Search and Filters: Allow product search by name, category, or barcode.

5. Technology Stack

Android POS System Requirements

Frontend:

- Language: Kotlin (preferred) or Java for Android development.

- Barcode Scanner: Google ML Kit or ZXing library.

- UI Framework: Jetpack Compose for modern UI design.

Backend:

- Database: SQLite for local storage.

Excel Handling:

- Use libraries like Apache POI or Google Sheets API for reading and writing Excel files.

6. Non-Functional Requirements

- Performance: The app should load within 2 seconds and handle large stock data efficiently.