Software Requirements Document (SRD)

Apartment Complex Account Management System

1. Introduction

1.1 Purpose

This document outlines the requirements for a **Simple Account Management System** designed for an apartment complex to manage income, expenses, and funds. The system will help track financial transactions, maintain account balances, and generate reports for better financial oversight.

1.2 Scope

The system will:

- Allow creation and management of different account types (Income, Expense, Assets, Cash, Bank, etc.).
- Record transactions with dates and default currency (INR).
- Track opening and closing balances for key accounts.
- Calculate monthly income, expenses, and profit/loss.
- Generate reports on account status, fund availability, and transaction history.
- Prevent transactions that would result in a negative balance in Cash or Bank accounts.

1.3 Definitions

- **Account**: A financial record (e.g., Rent Income, Maintenance Expense, Cash Account).
- **Transaction**: A financial entry (income or expense) recorded against an account.
- **Opening Balance**: The initial amount in an account at the start of a period.
- **Closing Balance**: The remaining amount in an account after transactions.

---

2. Functional Requirements

2.1 Account Management

2.1.1 Create an Account

- The system shall allow users to create accounts with:
  - **Account Name** (e.g., "Rent Income," "Maintenance Expense").
  - **Account Type** (Income, Expense, Asset, Liability, Cash, Bank).
  - **Opening Balance** (default: 0).
  - **Description** (optional).
- Example of all accounts

A.Income Accounts

- **Rent Income** (Monthly rent from tenants)
- **Maintenance Fees** (Recurring charges for upkeep)
- **Late Payment Penalties** (Fines for delayed rent)
- **Parking Fees** (Income from parking spaces)
- **Utility Reimbursements** (Tenant payments for shared utilities)
- **Security Deposit Forfeitures** (If a tenant breaches contract)
- **Guest Room Booking Income** (If the complex offers temporary stays)
- **Community Hall booking Income**
- **Income from different events**
- **Income due to Bank interests**

B. Expense Accounts

- **Electricity Bill** (Common area electricity)
- **Generator services including Fuel**
- **Water Bill** (Building water supply)
- **Staff Salaries** (Security, Housekeeping, managers)
- **Repairs & Maintenance** (Plumbing, painting, etc.)
- **Garbage Collection Fees**

- **Property Taxes**
- **Insurance Premiums** (Building insurance)
- **Internet & Cable** (Common area services)
- **Intercom Services**
- **CCTV Services**
- **Elevator Maintenance**
- **Gardening/Landscaping**
- **Fire Safety Equipment** (Extinguishers, alarms)
- **Legal & Professional Fees** (Lawyer, accountant)
- **Advertising & Marketing** (Vacancy promotions)
- **Housekeeping Consumables**
- **Office Room Consumables**
- **Expense due to cultural festivals**

3. Asset Accounts

- **Cash in Hand** (Physical cash available)
- **Bank Account - SBI** (Primary operating account)
- **Fixed Deposits** (Earning interest)
- **Building Fund Reserve** (Long-term savings)
- **Furniture & Appliances** (Common area assets)
- **Security Deposits Held** (Refundable tenant deposits)

4. Liability Accounts

- **Security Deposit Liability** (Owed back to tenants)
- **Pending Vendor Payments** (Unpaid bills)
- **Loan Payable** (If the complex has a bank loan)
- **Advance Maintenance Fees** (Prepaid by tenants)

4. Receivable Accounts

- **Rent Receivable** (Unpaid rent from tenants)

- **Parking Fee Receivable** (Unpaid parking charges)
- **Damage Charge Receivable** (If tenants owe for property damage)
- **Security Deposit Refund Receivable** (If a contractor owes money back to the complex)

2.1.2 Update an Account

- The system shall allow users to modify:
  - Account Name, Type, and Description.
  - Opening Balance (with validation to prevent incorrect updates).

2.1.3 View Accounts

- The system shall display a list of all accounts with:
  - Account Name, Type, Current Balance.
  - Filtering by Account Type (e.g., view only Cash/Bank accounts).

2.2 Transaction Management

2.2.1 Record a Transaction

- The system shall allow users to record transactions with:
  - **Date** (default: current date).
  - **Account** (select from existing accounts).
  - **Amount** (positive for income, negative for expense).
  - **Description** (e.g., "July Rent Collected").
  - **Reference Number** (optional, for tracking).
- Transaction entry should be enabled as journal management system, should do debit and credit right accounts based on transaction type. For income and expenses related asset accounts like Cash or Bank should be updated. So transaction mode should use cash or bank respectively.

2.2.2 Prevent Negative Balances

- The system shall **block transactions** if:
  - A withdrawal from Cash/Bank exceeds the available balance.

    - An expense entry would make the balance negative.
- The system shall display an error: **"Insufficient funds. Transaction cannot be processed."**

2.3 Balance Tracking

2.3.1 Opening and Closing Balances

- The system shall maintain:
    - **Opening Balance** (carried from the previous period).
    - **Closing Balance** (updated after each transaction).

2.3.2 Monthly Profit/Loss Calculation

- The system shall calculate:
    - **Total Income** (sum of all income transactions in a month).
    - **Total Expenses** (sum of all expense transactions in a month).
    - **Net Profit/Loss** = (Total Income – Total Expenses).

2.4 Reporting

2.4.1 Account Status Report

- The system shall generate a report showing:
    - Current balance of each account.
    - Filter by date range.

2.4.2 Fund Status Report

- The system shall display:
    - **Cash Balance** (total available cash).
    - **Bank Balance** (total available in bank accounts).

2.4.3 Transaction History

- The system shall list all transactions with:
    - Date, Account, Amount, Description.

    - Sorting by date (newest first).
    - Filtering by account or date range.

---

3. Non-Functional Requirements

3.1 Performance

- The system should load account balances and reports within **2 seconds**.
- Support at least **1000 transactions per month** without performance degradation.

3.2 Security

- Only **authorized users** can modify accounts or transactions.
- All transactions should be **logged** for audit purposes.

3.3 Usability

- Simple, intuitive interface with dropdowns for account selection.
- Error messages should be clear (e.g., "Invalid amount entered").

3.4 Data Integrity

- Transactions once recorded **cannot be deleted** (only flagged as "voided" if needed).
- All calculations (balances, profit/loss) must be **accurate and consistent**.

---

4. Technical Specifications

4.1 Technology Stack

- **Backend**: Python (Django/Flask) or Node.js.
- **Database**: SQLite (for small scale) / PostgreSQL (for scalability).

- **Frontend**: HTML, CSS, JavaScript (or a lightweight framework like React).

---

5. Future Enhancements (Optional)

- **Multi-Currency Support** (USD, EUR).
- **Automated Bank Sync** (via APIs).
- **User Roles** (Admin, Accountant, Viewer).

---

6. Conclusion

This system will provide the apartment complex with a **simple yet effective** way to manage finances, track transactions, and ensure funds are properly maintained without overdrafts.