Account Management User Stories

User Story ID: US-001

- **Title:** Create a New Financial Account
- **As an** Account Manager,
- **I want to** create different types of financial accounts (like Income, Expense, Asset, etc.),
- **so that I** can accurately categorize and track the apartment complex's finances.
- **Priority:** High
- **Acceptance Criteria:**
  - The system must allow the creation of an account with an Account Name, Account Type (Income, Expense, Asset, Liability, Cash, Bank), and an optional Description.
  - The opening balance for a new account must default to 0.
  - The user must be able to select the Account Type from a predefined list.

User Story ID: US-002

- **Title:** Update Existing Account Details
- **As an** Account Manager,
- **I want to** modify the name, type, and description of an existing account,
- **so that I** can keep the account information current and correct any errors.
- **Priority:** Medium
- **Acceptance Criteria:**
  - The system must allow updating the Account Name, Type, and Description for any existing account.
  - The system should allow for the modification of the opening balance, but only with proper validation to prevent incorrect financial entries.

User Story ID: US-003

- **Title:** View a List of All Accounts
- **As an** Account Manager,
- **I want to** view a comprehensive list of all created accounts,
- **so that I** can get a quick overview of all financial categories.
- **Priority:** Medium
- **Acceptance Criteria:**
  - The system must display a list showing the Account Name, Type, and its Current Balance.
  - The user must be able to filter the list to show only accounts of a specific type (e.g., view only Bank accounts).

Transaction Management User Stories

User Story ID: US-004

- **Title:** Record a Financial Transaction
- **As an** Account Manager,
- **I want to** record income and expense transactions against specific accounts,
- **so that I** can maintain an accurate log of all financial activities.
- **Priority:** High
- **Acceptance Criteria:**

    - When recording a transaction, the user must be able to input a Date (defaults to the current date), select an Account, enter an Amount, and add a Description.
    - The system should correctly handle debits and credits; for instance, an expense transaction should correctly update the balance of the linked Cash or Bank account.
    - An optional Reference Number field must be available for tracking purposes.

User Story ID: US-005

- **Title:** Prevent Overdrafts in Cash and Bank Accounts
- **As an** Account Manager,
- **I want the** system to prevent me from recording an expense that exceeds the available balance in a Cash or Bank account,
- **so that I** can avoid overdrawing funds and maintain financial integrity.
- **Priority:** High
- **Acceptance Criteria:**
    - The system must block any transaction that would result in a negative balance for any Cash or Bank account.
    - If a transaction is blocked due to insufficient funds, the system must display a clear error message like: "Insufficient funds. Transaction cannot be processed."

Reporting & Tracking User Stories

User Story ID: US-006

- **Title:** Generate an Account Status Report
- **As an** Account Manager,
- **I want to** generate a report showing the current balance of all accounts,
- **so that I** can review the financial standing of the apartment complex at a glance.
- **Priority:** Medium
- **Acceptance Criteria:**
    - The report must show the current balance for each account.
    - The user must be able to filter the report by a specific date range.
    - The report should load within 2 seconds.

User Story ID: US-007

- **Title:** View Total Cash and Bank Balances
- **As an** Account Manager,
- **I want to** view a fund status report,
- **so that I** can quickly see the total amount of liquid funds available in cash and in the bank.
- **Priority:** Medium
- **Acceptance Criteria:**
    - The report must clearly display the total balance of all accounts designated as 'Cash in Hand'.
    - The report must also display the combined total balance of all accounts designated as 'Bank Account'.

User Story ID: US-008

- **Title:** Review Transaction History

- **As an** Account Manager,
- **I want to** view a detailed history of all transactions,
- **so that I** can audit financial records and track specific entries over time.
- **Priority:** Medium
- **Acceptance Criteria:**
    - The transaction history must list the Date, Account, Amount, and Description for each transaction.
    - The list must be sortable by date, with the newest transactions appearing first by default.
    - The user must be able to filter the transaction history by a specific account or a date range.

User Story ID: US-009

- **Title:** Calculate Monthly Profit and Loss
- **As an** Account Manager,
- **I want the** system to automatically calculate the net profit or loss for each month,
- **so that I** can assess the monthly financial performance of the apartment complex.
- **Priority:** Low
- **Acceptance Criteria:**
    - The system must calculate the total income by summing all income transactions within a given month.
    - The system must calculate total expenses by summing all expense transactions within the same month.
    - The system must display the Net Profit/Loss, calculated as (Total Income - Total Expenses).

Based on your request, here are the expanded scenarios for the user story **US-004: Record a Financial Transaction**.

User Story: US-004 - Record a Financial Transaction

- **As an** Account Manager,
- **I want to** record income and expense transactions against specific accounts,
- **so that I** can maintain an accurate log of all financial activities.

Detailed Scenarios

- **Happy Path (Income Transaction):** The user selects an income account (e.g., "Rent Income") and the corresponding asset account (e.g., "Bank Account - SBI"). They enter the date, a valid amount, and a description. The system validates the data, records the transaction, and correctly increases the balances of both the income and bank accounts.
- **Happy Path (Expense Transaction):** The user selects an expense account (e.g., "Staff Salaries") and the asset account to pay from (e.g., "Bank Account - SBI"). The available balance is sufficient to cover the expense. The system validates the data, processes the payment, and correctly updates the balances, decreasing the bank account balance.
- **Edge Case (Future-Dated Transaction):** The user enters a date in the future to schedule a known upcoming expense. The system should accept and record the transaction with the future date.

- **Error Handling (Insufficient Funds):** The user attempts to record an expense for an amount that is greater than the current balance of the selected Cash or Bank account. The system must block the transaction and display a clear error message stating, "Insufficient funds. Transaction cannot be processed".
- **Error Handling (Invalid Data Input):** The user enters non-numeric text into the "Amount" field or fails to select an account from the dropdown. The system must prevent the transaction from being saved and display an inline error message guiding the user to correct the input, such as "Invalid amount entered".

Updated Acceptance Criteria

- **Given** I am on the "Record Transaction" screen,
  - **When** I select an income account, an asset account (Cash/Bank), and enter a valid positive amount,
  - **Then** the system must successfully record the transaction and the balances of both affected accounts must be updated correctly.
- **Given** I am recording an expense from a "Bank Account",
  - **When** the expense amount is less than or equal to the available bank balance,
  - **Then** the transaction must be processed successfully, and the bank balance must be reduced by the transaction amount.
- **Given** I am recording an expense from a "Cash" or "Bank" account,
  - **When** the expense amount is greater than the available balance in that account,
  - **Then** the system must prevent the transaction and show the error: "Insufficient funds. Transaction cannot be processed".
- **Given** I am entering transaction details,
  - **When** I enter non-numeric data in the "Amount" field or leave a required field (like "Account" or "Amount") blank,
  - **Then** the system must display a clear validation message and prevent the transaction from being saved until the error is corrected.
- **Given** I am recording a transaction,
  - **When** I enter a date in the future,
  - **Then** the system must accept the transaction and log it with the specified future date.
- ***Requirement Traceability Matrix (RTM)***
- *This matrix traces the business requirements outlined in the Software Requirements Document to their corresponding user stories and subsequent development and testing artifacts.*

Req ID	User Story ID	Description	Design Ref	Dev Task	Test Case
BR-01	US-001	Create a New Financial Account	DD-010	DEV-201	TC-301
BR-02	US-002	Update Existing Account Details	DD-011	DEV-202	TC-302
BR-03	US-003	View and Filter a List of All Accounts	DD-012	DEV-203	TC-303
BR-04	US-004	Record a Financial Transaction	DD-020	DEV-204	TC-304

BR-05	US-005	Prevent Overdrafts in Cash and Bank Accounts	DD-021	DEV-205	TC-305
BR-06	US-006	Generate an Account Status Report	DD-030	DEV-206	TC-306
BR-07	US-007	View Total Cash and Bank Balances	DD-031	DEV-207	TC-307
BR-08	US-008	Review and Filter Transaction History	DD-032	DEV-208	TC-308
BR-09	US-009	Calculate Monthly Profit and Loss	DD-033	DEV-209	TC-309

Here is a validation review of the user stories, identifying gaps in the acceptance criteria based on the non-functional requirements in the provided document.

Completeness and Gap Analysis Feedback 🧐

Based on a review of the generated user stories against the project's non-functional requirements, here are the identified gaps and suggestions for improvement:

- **Gap in Data Integrity for Transactions:**
  - **Observation:** The current acceptance criteria for **US-004 (Record a Financial Transaction)** does not include the critical data integrity rule that "Transactions once recorded cannot be deleted (only flagged as 'voided' if needed)".
  - **Recommendation:** A new user story should be created for "**Voiding a Transaction**" to ensure financial records cannot be permanently deleted, thus maintaining a clear audit trail.
- **Missing Security and Authorization Criteria:**
  - **Observation:** The user stories for modifying data, such as **US-002 (Update Account)** and **US-004 (Record Transaction)**, are missing acceptance criteria related to authorization. The requirements explicitly state, "Only authorized users can modify accounts or transactions".
  - **Recommendation:** Add an acceptance criterion to all relevant user stories (e.g., US-002, US-004) to **verify the user's authorization level** before allowing any modification or creation of financial data.
- **Missing Performance Requirements in Reporting:**
  - **Observation:** The user stories for generating reports, specifically **US-006 (Generate Account Status Report)** and **US-008 (Review Transaction History)**, do not incorporate the specified performance requirement. The system must "load account balances and reports within 2 seconds".
  - **Recommendation:** Add a new acceptance criterion to both US-006 and US-008 to state that the **report data must be generated and displayed to the user in under 2 seconds**.
- **Implicit Audit Logging:**
  - **Observation:** The requirement that "All transactions should be logged for audit purposes" is a backend process not fully captured in the user-facing stories.

- **Recommendation:** Add a technical acceptance criterion to **US-004 (Record Transaction)** ensuring that every new transaction entry **creates an immutable log entry** with a timestamp and user ID for security auditing.