CSE305 Software Engineering - Domain Model

Student: Başar Orhanbulucu
Student ID: 221805031
Project: Personal Asset Tracking System (Kişisel Varlık Takip Sistemi)
Methodology: Agile
Platform: Mobile

1. Project Overview

1.1. Purpose and Goal

The primary goal of this project is to develop a mobile application, "Personal Asset Tracking System." The system's core focus is on **personal budgeting**, **expense tracking**, and **savings goal management**.

The application will provide an intuitive interface for users to log their daily income and expenses, categorize their spending, and track progress toward personal financial goals. A key design principle is to provide a solid foundation for future expansion into comprehensive asset management (e.g., stocks, real estate) while keeping the initial MVP simple and focused.

1.2. Scope Definition

The Minimum Viable Product (MVP) will focus on budgeting and goals.

1. **User Authentication:** Secure registration and login.
2. **Asset/Account Management:** CRUD operations for *cash-based* assets (e.g., "Bank Account", "Wallet"). Balances will be **manually updated** by the user to maintain simplicity. The data structure will be generic to support other asset types in the future.
3. **Transaction Management:** Full CRUD for logging 'Income' and 'Expense' transactions, with categorization, search, and filtering.
4. **Goal Management:** CRUD for setting and tracking various financial goals (e.g., savings targets, expense limits).
5. **Dashboard:** A summary screen with visualizations for cash flow and goal progress.

1.3. Technology Stack

- **Frontend (Mobile):** Flutter (using Dart language).
- **Backend (BaaS):** Firebase
  - **Authentication:** Firebase Authentication.
  - **Database:** Cloud Firestore (NoSQL).

2. Problem Domain Description

The system centers on the **User**. A **User** must register and authenticate to access their private financial data.

The **User** manages their finances by logging **Transactions**. Each **Transaction** is a record with an amount, a date, a type ('Income' or 'Expense'), and a categoryName (e.g., 'Groceries', 'Salary'). The system will support grouping these transactions by month, and provide search and filter capabilities on the transaction history screen.

To track their net worth, the **User** can create and manage one or more **Assets**. For the MVP, these **Assets** will represent simple *cash accounts* (e.g., 'Bank Account', 'Wallet'). Each **Asset** has a name and a currentValue (balance). A key feature for MVP simplicity is that this currentValue is **updated manually** by the user. It is *not* automatically calculated from transactions. This design simplifies the logic immensely but provides the flexibility to add an optional automatic link in the future.

Finally, the **User** can set personal financial **Goals**. This is a flexible entity. A **Goal** has a name, a targetAmount, a currentAmount, and a goalType ('Savings', 'ExpenseLimit', 'IncomeTarget').

- A 'Savings' goal (e.g., 'Buy Headphones') tracks currentAmount rising toward targetAmount.
- An 'ExpenseLimit' goal (e.g., 'Limit Fast Food') tracks currentAmount (total spending in that category) and warns if it approaches the targetAmount limit.
- An 'IncomeTarget' goal (e.g., 'Earn 500 in Freelance') tracks currentAmount (total income in that category) rising toward targetAmount.
  This flexible model allows users to set goals for saving, for limiting expenses, or for achieving income targets, all using one class. A User is not required to have any goals.

3. Domain Model (Conceptual Classes and Relationships)

Class: User (Kullanıcı)

The primary actor. All other data is owned by a User.

- **Attributes:**
  - userId (PK - From Firebase Auth)
  - email (Unique, for login)
  - firstName, lastName
  - registrationDate
- **Relationships:**
  - 1-to-Many (1..*) -> **Asset** (A User can have multiple assets).

    - 1-to-Many (1..*) -> **Transaction** (A User logs many transactions).
    - 1-to-Many (0..*) -> **Goal** (A User can have zero or more goals).

Class: Asset (Varlık)

Represents a source of value. For the MVP, this will only be used for cash accounts, but is designed for future expansion.

- **Attributes:**
    - assetId (PK - Auto-generated)
    - name (e.g., "Akbank Hesabım", "Cüzdanım")
    - assetType (e.g., "Cash", "Stock", "RealEstate" - *For MVP, only "Cash" will be used*)
    - currentValue (The current balance/value, **manually updated by user**)
    - iconName (Optional, for UI)
- **Relationships:**
    - Many-to-One (*..1) -> **User** (Belongs to one User).

Class: Transaction (İşlem)

The core entity for budgeting; a single income or expense record.

- **Attributes:**
    - transactionId (PK - Auto-generated)
    - amount (The monetary value)
    - type (Enum: 'Income' | 'Expense')
    - categoryName (String, e.g., "Maaş", "Market", "Faturalar", "KYK Kredisi")
    - date (Timestamp of the transaction)
    - description (Optional, user notes)
- **Relationships:**
    - Many-to-One (*..1) -> **User** (Belongs to one User).
    - *(Future Relationship): Can be optionally linked to an Asset.*

Class: Goal (Hedef)

A flexible entity for tracking financial targets (savings, expense limits, or income goals).

- **Attributes:**
    - goalId (PK - Auto-generated)
    - name (e.g., "Yeni Kulaklık", "Aylık Harcama Limiti")
    - goalType (Enum: 'Savings' | 'ExpenseLimit' | 'IncomeTarget')
    - targetAmount (The total amount to save / The limit / The target income)
    - currentAmount (The amount saved so far / Spent so far / Earned so far)
    - startDate, endDate (Optional, for time-bound goals)
- **Relationships:**
    - Many-to-One (*..1) -> **User** (Belongs to one User).

Domain Model