Personal Finance Management System

Product Requirements Document (PRD)

Version 1.0 | May 2025

Table of Contents

1. Executive Summary

The Personal Finance Management System is a comprehensive web application designed to help users manage their financial accounts, track transactions, and monitor their financial health. The system supports multiple account types including savings accounts, credit cards, and investment accounts, with features for transaction management, payee management, and detailed financial reporting.

2. Product Overview

2.1 Product Vision

To provide users with a unified platform to manage all their financial accounts, enabling better financial decision-making through comprehensive tracking, categorization, and analysis of their financial data.

2.2 Target Users

- **Primary:** Individual consumers managing personal finances
- **Secondary:** Small business owners tracking business expenses
- **Tertiary:** Financial advisors managing client portfolios

2.3 Key Benefits

- Centralized financial account management
- Real-time transaction tracking and categorization
- Investment portfolio monitoring
- Automated payee management
- Comprehensive financial reporting

3. User Stories & Requirements

Epic 1: Account Management
As a user, I want to manage multiple financial accounts in one place so that I can have a comprehensive view of my finances.

Acceptance Criteria:

- User can add savings accounts, credit cards, and investment accounts
- Each account displays current balance and recent activity
- Account details can be edited and updated
- Accounts can be categorized and organized

Epic 2: Transaction Management
As a user, I want to track and categorize all my transactions so that I can understand my spending patterns.

Acceptance Criteria:

- Transactions can be manually added or imported

- Each transaction includes date, amount, description, and category
- Transactions can be searched and filtered
- Transaction history is maintained with audit trail

Epic 3: Investment Tracking
As an investor, I want to monitor my investment portfolio performance so that I can make informed investment decisions.

Acceptance Criteria:

- Investment accounts show current value and performance metrics
- Portfolio allocation is displayed visually
- Historical performance tracking
- Individual security performance monitoring

Epic 4: Payee Management
As a user, I want to manage my payees and payment recipients so that I can efficiently process payments and track expenses.

Acceptance Criteria:

- Payees can be added, edited, and deleted
- Payee information includes name, address, and payment details
- Transaction history per payee is available
- Payee categorization for expense tracking

4. Feature Specifications

Dashboard

- Account overview with balances
- Recent transaction summary
- Quick action buttons

- Financial health indicators

Account Management

- Add/edit/delete accounts
- Account type configuration
- Balance tracking
- Account settings and preferences

Transaction Register

- Transaction entry and editing
- Category assignment
- Search and filtering
- Bulk operations

Investment Tracking

- Portfolio overview
- Security performance
- Asset allocation charts
- Dividend tracking

Payee Management

- Payee directory
- Payment history
- Recurring payment setup
- Vendor categorization

Reporting

- Spending analysis
- Income vs expenses
- Investment performance
- Custom report builder

5. Database Schema

5.1 Core Tables

users

```
id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)
email (VARCHAR(255), UNIQUE, NOT NULL)
password_hash (VARCHAR(255), NOT NULL)
first_name (VARCHAR(100), NOT NULL)
last_name (VARCHAR(100), NOT NULL)
created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)
updated_at (TIMESTAMP, ON UPDATE CURRENT_TIMESTAMP)
is_active (BOOLEAN, DEFAULT TRUE)
```

account_types

```
id (INT, PRIMARY KEY, AUTO_INCREMENT)
name (VARCHAR(50), NOT NULL) -- 'Savings', 'Credit
Card', 'Investment'
description (TEXT)
icon (VARCHAR(50))
is_active (BOOLEAN, DEFAULT TRUE)
```

accounts

id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)

user_id (BIGINT, FOREIGN KEY → users.id)

account_type_id (INT, FOREIGN KEY → account_types.id)

account_name (VARCHAR(200), NOT NULL)

account_number (VARCHAR(50))

institution_name (VARCHAR(200))

current_balance (DECIMAL(15,2), DEFAULT 0.00)

available_balance (DECIMAL(15,2))

credit_limit (DECIMAL(15,2)) -- For credit cards

interest_rate (DECIMAL(5,4))

opening_date (DATE)

is_active (BOOLEAN, DEFAULT TRUE)

created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)

updated_at (TIMESTAMP, ON UPDATE CURRENT_TIMESTAMP)

transaction_categories

id (INT, PRIMARY KEY, AUTO_INCREMENT)

user_id (BIGINT, FOREIGN KEY → users.id)

name (VARCHAR(100), NOT NULL)

parent_category_id (INT, FOREIGN KEY → transaction_categories.id)

color (VARCHAR(7)) -- Hex color code

icon (VARCHAR(50))

is_system (BOOLEAN, DEFAULT FALSE)

is_active (BOOLEAN, DEFAULT TRUE)

transactions

id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)

account_id (BIGINT, FOREIGN KEY → accounts.id)

category_id (INT, FOREIGN KEY → transaction_categories.id)

payee_id (BIGINT, FOREIGN KEY → payees.id)

transaction_date (DATE, NOT NULL)

amount (DECIMAL(15,2), NOT NULL)

description (VARCHAR(500))

reference_number (VARCHAR(100))

transaction_type (ENUM('DEBIT', 'CREDIT'), NOT NULL)

status (ENUM('PENDING', 'CLEARED', 'RECONCILED'), DEFAULT 'PENDING')

notes (TEXT)

created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)

updated_at (TIMESTAMP, ON UPDATE CURRENT_TIMESTAMP)

payees

id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)

user_id (BIGINT, FOREIGN KEY → users.id)

name (VARCHAR(200), NOT NULL)

address_line1 (VARCHAR(200))

address_line2 (VARCHAR(200))

city (VARCHAR(100))

state (VARCHAR(50))

zip_code (VARCHAR(20))

country (VARCHAR(100))

phone (VARCHAR(20))

email (VARCHAR(255))

website (VARCHAR(255))

default_category_id (INT, FOREIGN KEY → transaction_categories.id)

is_active (BOOLEAN, DEFAULT TRUE)

created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)

updated_at (TIMESTAMP, ON UPDATE CURRENT_TIMESTAMP)

5.2 Investment Tables

securities

id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)

symbol (VARCHAR(20), UNIQUE, NOT NULL)

name (VARCHAR(200), NOT NULL)

security_type (ENUM('STOCK', 'BOND', 'MUTUAL_FUND', 'ETF', 'OPTION', 'OTHER'))

exchange (VARCHAR(10))

sector (VARCHAR(100))

current_price (DECIMAL(15,4))

last_updated (TIMESTAMP)

is_active (BOOLEAN, DEFAULT TRUE)

holdings

id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)

account_id (BIGINT, FOREIGN KEY → accounts.id)

security_id (BIGINT, FOREIGN KEY → securities.id)

quantity (DECIMAL(15,6), NOT NULL)

average_cost (DECIMAL(15,4))

market_value (DECIMAL(15,2))

last_updated (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP)

investment_transactions

id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)

account_id (BIGINT, FOREIGN KEY → accounts.id)

security_id (BIGINT, FOREIGN KEY → securities.id)

transaction_date (DATE, NOT NULL)

transaction_type (ENUM('BUY', 'SELL', 'DIVIDEND', 'SPLIT', 'TRANSFER_IN', 'TRANSFER_OUT'))

quantity (DECIMAL(15,6))

price (DECIMAL(15,4))

total_amount (DECIMAL(15,2))

fees (DECIMAL(15,2), DEFAULT 0.00)

description (VARCHAR(500))

created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)

5.3 Supporting Tables

recurring_transactions

id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)

account_id (BIGINT, FOREIGN KEY → accounts.id)

category_id (INT, FOREIGN KEY → transaction_categories.id)

payee_id (BIGINT, FOREIGN KEY → payees.id)

amount (DECIMAL(15,2), NOT NULL)

description (VARCHAR(500))

frequency (ENUM('DAILY', 'WEEKLY', 'MONTHLY', 'QUARTERLY', 'YEARLY'))

start_date (DATE, NOT NULL)

end_date (DATE)

next_due_date (DATE)

is_active (BOOLEAN, DEFAULT TRUE)

created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)

budgets

```
id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)
user_id (BIGINT, FOREIGN KEY → users.id)
category_id (INT, FOREIGN KEY →
transaction_categories.id)
budget_period (ENUM('MONTHLY', 'QUARTERLY',
'YEARLY'))
budget_amount (DECIMAL(15,2), NOT NULL)
start_date (DATE, NOT NULL)
end_date (DATE, NOT NULL)
is_active (BOOLEAN, DEFAULT TRUE)
created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)
```

account_balances_history

```
id (BIGINT, PRIMARY KEY, AUTO_INCREMENT)
account_id (BIGINT, FOREIGN KEY → accounts.id)
balance_date (DATE, NOT NULL)
balance (DECIMAL(15,2), NOT NULL)
created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)
INDEX idx_account_date (account_id, balance_date)
```

6. Technical Requirements

6.1 Performance Requirements

- Page load times under 2 seconds
- Support for 10,000+ transactions per account
- Real-time balance updates
- Responsive design for mobile and desktop

6.2 Security Requirements

- Multi-factor authentication
- Encrypted data storage
- Session management and timeout

- Audit logging for all financial transactions
- PCI DSS compliance for payment processing

6.3 Integration Requirements

- Bank API integration for account synchronization
- Investment data feed integration
- Export capabilities (CSV, PDF, Excel)
- Backup and restore functionality

7. Success Metrics

7.1 User Engagement

- Daily active users > 70% of registered users
- Average session duration > 10 minutes
- Transaction entry completion rate > 95%

7.2 System Performance

- System uptime > 99.5%
- API response time < 500ms
- Database query performance optimization

7.3 Business Metrics

- User retention rate > 80% after 30 days
- Customer satisfaction score > 4.5/5
- Support ticket resolution time < 24 hours

Document Version: 1.0 | **Date:** May 27, 2025

Status: Ready for Development | **Next Review:** June 15, 2025