家計簿アプリ - 要件定義書

1. プロジェクト概要

React/Express/PostgreSQLスタックを使用した家計簿管理アプリケーション

2. 機能要件

2.1 収支管理機能

- **収入の記録**
  - 金額、日付、メモの入力
  - 記録の編集・削除
- **支出の記録**
  - 金額、日付、カテゴリ、メモの入力
  - 記録の編集・削除

2.2 カテゴリ管理機能

- 支出カテゴリの管理
  - カテゴリの追加・編集・削除
  - 基本カテゴリ：食費、交通費、光熱費、娯楽費、日用品、医療費、その他

2.3 表示機能

- 月別収支一覧表示
- 日別収支一覧表示
- 現在の残高表示
- カテゴリ別支出合計表示

3. ユーザーストーリー

収支記録系

- ユーザーとして、収入を記録したい（金額、日付、メモ）
- ユーザーとして、記録した収入を修正・削除したい
- ユーザーとして、支出を記録したい（金額、日付、カテゴリ、メモ）
- ユーザーとして、記録した支出を修正・削除したい

カテゴリ管理系

- ユーザーとして、支出カテゴリを追加したい
- ユーザーとして、既存のカテゴリを編集したい
- ユーザーとして、使わないカテゴリを削除したい

表示・確認系

- ユーザーとして、今月の収支一覧を見たい
- ユーザーとして、指定した月の収支を見たい
- ユーザーとして、現在の残高を確認したい
- ユーザーとして、カテゴリ別の支出合計を見たい

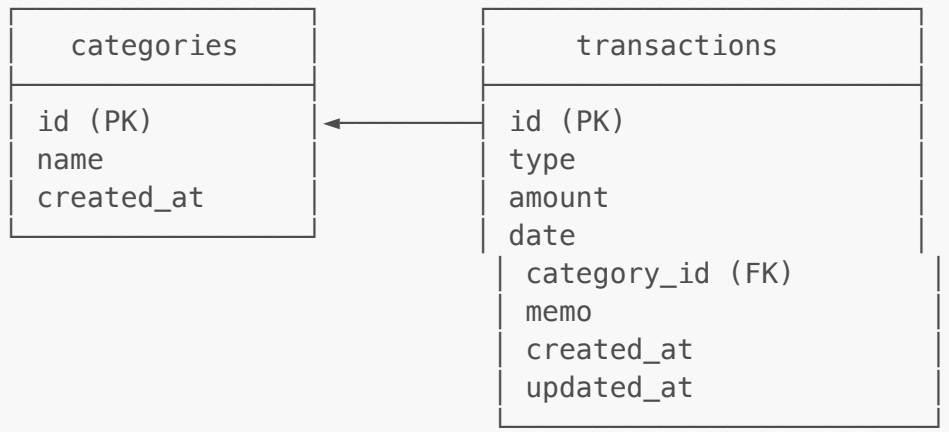
4. データモデル

4.1 テーブル構成

```sql
-- カテゴリテーブル
CREATE TABLE categories (
    id          SERIAL PRIMARY KEY,
    name        VARCHAR(50) NOT NULL,
    created_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- 取引記録テーブル
CREATE TABLE transactions (
    id          SERIAL PRIMARY KEY,
    type        VARCHAR(10) NOT NULL,  -- 'income' or 'expense'
    amount      DECIMAL(10,2) NOT NULL,
    date        DATE NOT NULL,
    category_id INTEGER REFERENCES categories(id),
    memo        TEXT,
    created_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

4.2 データベース設計図



5. 技術スタック

5.1 フロントエンド

- **React 18** - メインフレームワーク

- **TypeScript** - 型安全性
- **CSS Modules** - スタイリング
- **React Router** - ページ遷移
- **Axios** - API通信

5.2 バックエンド

- **Express.js** - Webサーバー
- **TypeScript** - 型安全性
- **pg** - PostgreSQLクライアント
- **cors** - CORS設定
- **dotenv** - 環境変数管理

5.3 データベース

- **PostgreSQL** - メインDB

5.4 開発環境

- **Node.js** - ランタイム
- **npm** - パッケージマネージャー
- **nodemon** - 開発時の自動再起動
- **concurrently** - フロント・バック同時起動

6. プロジェクト構造

```
household-budget-app/
├── client/                    # React フロントエンド
│   ├── public/
│   ├── src/
│   │   ├── components/        # 共通コンポーネント
│   │   ├── pages/             # ページコンポーネント
│   │   ├── hooks/             # カスタムフック
│   │   ├── services/          # API通信
│   │   ├── types/             # TypeScript型定義
│   │   └── App.tsx
│   ├── package.json
│   └── tsconfig.json
├── server/                    # Express バックエンド
│   ├── src/
│   │   ├── controllers/       # コントローラー
│   │   ├── models/            # データモデル
│   │   ├── routes/            # ルーティング
│   │   ├── database/          # DB接続・マイグレーション
│   │   └── app.ts
│   ├── package.json
│   └── tsconfig.json
├── database/                  # SQL関連
│   ├── migrations/            # マイグレーションファイル
│   └── seeds/                # 初期データ
```

```
├── package.json              # ルートレベル（開発用）
└── README.md
```

7. 非機能要件

7.1 データ管理

- データの永続化（PostgreSQL）
- データの整合性確保

7.2 ユーザビリティ

- 直感的な操作画面
- レスポンシブデザイン

8. 開発フェーズ

フェーズ1: 環境セットアップ

- Node.js、PostgreSQL環境構築
- プロジェクト初期化

フェーズ2: データベース構築

- テーブル作成
- 初期データ投入

フェーズ3: バックエンド実装

- API作成
- CRUD操作実装

フェーズ4: フロントエンド実装

- React画面作成
- API連携

フェーズ5: 動作確認・テスト

- 結合テスト
- UI/UX調整

作成日: 2025年7月5日
作成者: 学習者向けプロジェクト