Money Clone - Low Level Design Document

1. System Overview

The Money Clone application is a personal finance management system that allows users to track multiple account types (Savings, Credit Card, Investment), manage transactions, payees, and categories. The system supports both desktop and mobile interfaces with responsive design.

2. Core User Stories

2.1 Authentication & Authorization

Epic: User Authentication System

US-AUTH-001: User Registration

- As a new user, I want to create an account so that I can access the financial management system
- Acceptance Criteria:
  - User can register with email and password
  - Email validation is required
  - Password must meet security requirements (min 8 chars, special chars, numbers)
  - Account activation via email verification
  - Duplicate email prevention

US-AUTH-002: User Login

- As a registered user, I want to log in securely so that I can access my financial data
- Acceptance Criteria:
  - Login with email/username and password
  - "Remember me" functionality
  - Account lockout after failed attempts
  - Password reset functionality
  - Session management with timeout

US-AUTH-003: Session Management

- As a user, I want my session to be secure and manageable
- Acceptance Criteria:
  - Auto-logout after inactivity
  - Secure session tokens
  - Multi-device session handling
  - Logout functionality

2.2 Account Management

Epic: Multi-Account Financial Management

US-ACC-001: Create Savings Account

- As a user, I want to create savings accounts so that I can track my savings
- Acceptance Criteria:
  - Account name (required)
  - Account number (optional)
  - Bank/Institution name
  - Initial balance
  - Account type selection (Savings)
  - Currency selection

US-ACC-002: Create Credit Card Account

- As a user, I want to create credit card accounts so that I can track my credit transactions
- Acceptance Criteria:
  - Card name/nickname (required)
  - Last 4 digits display
  - Credit limit setting
  - Current balance
  - Due date tracking
  - Minimum payment amount

US-ACC-003: Create Investment Account

- As a user, I want to create investment accounts so that I can track my portfolio
- Acceptance Criteria:
  - Account name (required)
  - Investment type (Stocks, Mutual Funds, etc.)
  - Current value
  - Initial investment amount
  - Portfolio tracking

US-ACC-004: View Account Summary

- As a user, I want to see all my accounts at a glance
- Acceptance Criteria:
  - Dashboard with all account types
  - Current balances
  - Recent transactions preview
  - Account status indicators
  - Quick action buttons

US-ACC-005: Account Details View

- As a user, I want to view detailed information about each account
- Acceptance Criteria:
  - Account information display
  - Transaction history
  - Account settings access
  - Edit account details
  - Account performance metrics (for investments)

2.3 Transaction Management

Epic: Transaction Processing System

US-TXN-001: Add Income Transaction

- As a user, I want to record income transactions so that I can track money coming in
- Acceptance Criteria:
  - Amount entry (positive)
  - Date selection (default: today)
  - Payee selection/creation
  - Category selection/creation
  - Description/memo field
  - Account selection
  - Transaction type: Income

US-TXN-002: Add Expense Transaction

- As a user, I want to record expense transactions so that I can track money going out
- Acceptance Criteria:
  - Amount entry (negative)
  - Date selection (default: today)
  - Payee selection/creation
  - Category selection/creation
  - Description/memo field
  - Account selection
  - Transaction type: Expense

US-TXN-003: Add Transfer Transaction

- As a user, I want to transfer money between accounts so that I can manage funds across accounts
- Acceptance Criteria:
  - From account selection
  - To account selection
  - Amount entry
  - Date selection
  - Description/memo field
  - Transaction type: Transfer
  - Automatic balance updates

US-TXN-004: Edit Transaction

- As a user, I want to modify existing transactions so that I can correct errors
- Acceptance Criteria:
  - Select transaction from list
  - Edit all transaction fields
  - Save changes with confirmation
  - Balance recalculation
  - Audit trail maintenance

US-TXN-005: Delete Transaction

- As a user, I want to delete transactions so that I can remove incorrect entries
- Acceptance Criteria:
  - Select transaction from list
  - Confirmation dialog
  - Balance recalculation
  - Soft delete for audit trail

US-TXN-006: View Transaction History

- As a user, I want to view my transaction history so that I can review my financial activity
- Acceptance Criteria:
  - Chronological transaction list
  - Filter by date range
  - Filter by account
  - Filter by category
  - Filter by payee
  - Search functionality
  - Pagination for large datasets

2.4 Payee Management

Epic: Payee Management System

US-PAY-001: Create Payee

- As a user, I want to create payees so that I can quickly select them for transactions
- Acceptance Criteria:
  - Payee name (required)
  - Default category assignment
  - Contact information (optional)
  - Notes field
  - Payee type (Person, Business, etc.)

US-PAY-002: Edit Payee

- As a user, I want to edit payee information so that I can keep it current
- Acceptance Criteria:
  - Modify all payee fields
  - Update default category
  - Save changes with confirmation
  - Impact on existing transactions (reference only)

US-PAY-003: Delete Payee

- As a user, I want to delete payees I no longer use
- Acceptance Criteria:
  - Confirmation dialog
  - Check for existing transaction references
  - Soft delete if referenced
  - Hard delete if not referenced

US-PAY-004: View Payee List

- As a user, I want to see all my payees in an organized list
- Acceptance Criteria:
  - Alphabetical sorting
  - Search functionality
  - Filter by payee type
  - Quick action buttons (Edit, Delete)
  - Usage statistics (transaction count)

US-PAY-005: Payee Auto-suggest

- As a user, I want payee suggestions while entering transactions
- Acceptance Criteria:
  - Auto-complete based on partial input
  - Most frequently used payees first
  - Create new payee option
  - Default category population

2.5 Category Management

Epic: Category Management System

US-CAT-001: Create Category

- As a user, I want to create categories so that I can organize my transactions
- Acceptance Criteria:
  - Category name (required)
  - Category type (Income, Expense, Transfer)
  - Parent category (for subcategories)
  - Color coding
  - Description field

US-CAT-002: Edit Category

- As a user, I want to edit categories so that I can refine my organization system
- Acceptance Criteria:
  - Modify category details
  - Change parent category
  - Update color coding
  - Impact on existing transactions (reference only)

US-CAT-003: Delete Category

- As a user, I want to delete unused categories
- Acceptance Criteria:
  - Confirmation dialog
  - Check for existing transaction references
  - Soft delete if referenced
  - Hard delete if not referenced
  - Handle subcategories appropriately

US-CAT-004: View Category Hierarchy

- As a user, I want to see my categories in a hierarchical structure
- Acceptance Criteria:
  - Tree view of categories and subcategories
  - Expand/collapse functionality
  - Drag-and-drop reordering
  - Usage statistics

US-CAT-005: Category Auto-suggest

- As a user, I want category suggestions based on payee selection
- Acceptance Criteria:
  - Auto-populate based on payee's default category
  - Learning from user behavior
  - Manual override capability

2.6 Responsive Design & Mobile Support

Epic: Cross-Platform User Experience

US-UI-001: Responsive Dashboard

- As a user, I want the dashboard to work well on all devices
- Acceptance Criteria:
  - Desktop: Multi-column layout with detailed information
  - Tablet: Adaptive layout with readable content
  - Mobile: Single-column stack with essential information
  - Touch-friendly controls on mobile
  - Consistent branding across devices

US-UI-002: Mobile Transaction Entry

- As a mobile user, I want a simplified transaction entry experience
- Acceptance Criteria:
  - Large, touch-friendly input fields
  - Simplified form with essential fields only
  - Quick amount entry (number pad)
  - Swipe gestures for navigation
  - Camera integration for receipt capture (future)

US-UI-003: Mobile Account Views

- As a mobile user, I want to easily view account information
- Acceptance Criteria:
  - Swipeable account cards
  - Essential information prominently displayed
  - Quick action buttons (Add Transaction, View Details)
  - Pull-to-refresh functionality

US-UI-004: Cross-Device Data Sync

- As a user, I want my data synchronized across all devices
- Acceptance Criteria:
  - Real-time data synchronization
  - Offline capability with sync on reconnection
  - Conflict resolution for simultaneous edits
  - Data consistency across devices

3. Technical Tasks

3.1 Database Design Tasks

TASK-DB-001: Design User Authentication Schema

- Create users table with fields: id, email, password_hash, created_at, updated_at, email_verified, last_login
- Create user_sessions table for session management
- Create password_reset_tokens table
- Define indexes for performance

TASK-DB-002: Design Account Schema

- Create accounts table with fields: id, user_id, account_type, name, account_number, institution, balance, currency, created_at, updated_at
- Create account_types lookup table
- Define foreign key relationships
- Create indexes for user_id and account_type

TASK-DB-003: Design Transaction Schema

- Create transactions table with fields: id, account_id, transaction_type, amount, date, payee_id, category_id, description, created_at, updated_at, deleted_at
- Create transaction_types lookup table
- Define foreign key relationships
- Create indexes for account_id, date, payee_id, category_id

TASK-DB-004: Design Payee Schema

- Create payees table with fields: id, user_id, name, default_category_id, contact_info, notes, payee_type, created_at, updated_at, deleted_at
- Create payee_types lookup table
- Define foreign key relationships
- Create indexes for user_id and name

TASK-DB-005: Design Category Schema

- Create categories table with fields: id, user_id, name, parent_id, category_type, color, description, created_at, updated_at, deleted_at
- Create category_types lookup table
- Define self-referencing foreign key for parent_id
- Create indexes for user_id and parent_id

3.2 Backend API Tasks

TASK-API-001: Implement Authentication Endpoints

- POST /api/auth/register - User registration
- POST /api/auth/login - User login
- POST /api/auth/logout - User logout
- POST /api/auth/refresh - Token refresh
- POST /api/auth/forgot-password - Password reset request
- POST /api/auth/reset-password - Password reset completion
- GET /api/auth/verify-email/:token - Email verification

TASK-API-002: Implement Account Management Endpoints

- GET /api/accounts - List user accounts
- POST /api/accounts - Create new account
- GET /api/accounts/:id - Get account details
- PUT /api/accounts/:id - Update account
- DELETE /api/accounts/:id - Delete account
- GET /api/accounts/:id/balance - Get current balance

TASK-API-003: Implement Transaction Management Endpoints

- GET /api/transactions - List transactions with filters
- POST /api/transactions - Create new transaction
- GET /api/transactions/:id - Get transaction details
- PUT /api/transactions/:id - Update transaction
- DELETE /api/transactions/:id - Delete transaction
- GET /api/transactions/search - Search transactions

TASK-API-004: Implement Payee Management Endpoints

- GET /api/payees - List user payees
- POST /api/payees - Create new payee
- GET /api/payees/:id - Get payee details
- PUT /api/payees/:id - Update payee
- DELETE /api/payees/:id - Delete payee
- GET /api/payees/search - Search payees with auto-suggest

TASK-API-005: Implement Category Management Endpoints

- GET /api/categories - List categories in hierarchy
- POST /api/categories - Create new category
- GET /api/categories/:id - Get category details
- PUT /api/categories/:id - Update category
- DELETE /api/categories/:id - Delete category
- GET /api/categories/search - Search categories

3.3 Frontend Development Tasks

TASK-FE-001: Implement Authentication Components

- Create Login component with form validation
- Create Registration component with email verification flow
- Create Password Reset components
- Implement protected route wrapper
- Create session timeout handler

TASK-FE-002: Implement Dashboard Components

- Create responsive dashboard layout
- Implement account summary cards
- Create recent transactions widget
- Implement quick action buttons
- Add loading states and error handling

TASK-FE-003: Implement Account Management Components

- Create account list view with responsive design
- Implement account creation forms for each type
- Create account details view with transaction history
- Implement account editing functionality
- Add account deletion with confirmation

TASK-FE-004: Implement Transaction Components

- Create transaction entry form with responsive design
- Implement transaction list with filtering and search
- Create transaction editing modal
- Implement transaction deletion with confirmation
- Add transaction import/export functionality

TASK-FE-005: Implement Payee Management Components

- Create payee list view with search and filter
- Implement payee creation and editing forms
- Create payee deletion with dependency checking
- Implement auto-suggest component for transaction entry
- Add payee usage statistics

TASK-FE-006: Implement Category Management Components

- Create hierarchical category tree view
- Implement category creation and editing forms
- Create drag-and-drop category reordering
- Implement category deletion with dependency checking
- Add category usage statistics and reporting

3.4 Error Handling & Validation Tasks

TASK-ERR-001: Implement API Error Handling

- Create standardized error response format
- Implement validation middleware for all endpoints
- Create error logging and monitoring
- Implement rate limiting and security measures
- Create user-friendly error messages

TASK-ERR-002: Implement Frontend Error Handling

- Create global error boundary component
- Implement form validation with real-time feedback
- Create error toast/notification system
- Implement offline state handling
- Create fallback UI components for errors

TASK-ERR-003: Implement Data Validation

- Create input sanitization for all user inputs
- Implement business logic validation (e.g., sufficient balance)
- Create data consistency checks
- Implement duplicate detection
- Create data integrity constraints

3.5 Performance & Security Tasks

TASK-PERF-001: Implement Performance Optimizations

- Create database query optimization
- Implement caching strategy for frequently accessed data
- Create lazy loading for large datasets
- Implement pagination for transaction lists
- Create image optimization for mobile

TASK-SEC-001: Implement Security Measures

- Create input sanitization and XSS prevention
- Implement CSRF protection
- Create secure session management
- Implement API rate limiting
- Create data encryption for sensitive information

4. Data Relationships

4.1 Entity Relationships

User → Accounts (1:N)

- One user can have multiple accounts
- Each account belongs to one user

User → Payees (1:N)

- One user can have multiple payees
- Each payee belongs to one user

User → Categories (1:N)

- One user can have multiple categories
- Each category belongs to one user

Account → Transactions (1:N)

- One account can have multiple transactions
- Each transaction belongs to one account

Payee → Transactions (1:N)

- One payee can be used in multiple transactions
- Each transaction has one payee

Category → Transactions (1:N)

- One category can be used in multiple transactions
- Each transaction has one category

Category → Category (1:N - Self-referencing)

- Categories can have parent-child relationships for hierarchy
- One parent category can have multiple subcategories

Payee → Category (N:1 - Default)

- Each payee can have a default category
- One category can be default for multiple payees

4.2 Business Rules

Transaction Rules:

- Transfer transactions must have both from_account and to_account
- Income transactions increase account balance
- Expense transactions decrease account balance
- Transfer transactions update both account balances
- Transaction amounts must be positive (sign determined by type)

Account Rules:

- Credit card accounts can have negative balances
- Savings and investment accounts typically have positive balances
- Account deletion requires zero balance or transaction transfer

Category Rules:

- Income categories can only be used with income transactions
- Expense categories can only be used with expense transactions
- Transfer categories are system-generated and not user-editable

Payee Rules:

- Payee names must be unique per user
- Payee deletion is soft delete if transactions exist
- Default categories are suggestions, not enforced

5. Screen Specifications

5.1 Dashboard Screen

Desktop Layout:

- Header with user menu and notifications
- Three-column layout: Account Summary | Recent Transactions | Quick Actions
- Account cards showing type, name, and current balance
- Mini transaction list (last 5 transactions)
- Quick add transaction button

Mobile Layout:

- Single column stacked layout
- Swipeable account cards
- Collapsible sections for transactions
- Floating action button for quick transaction entry

5.2 Account Detail Screen

Desktop Layout:

- Account information header with edit button
- Transaction list with filters and search
- Side panel with account statistics and actions

Mobile Layout:

- Account header with key information
- Tab navigation for transactions, details, and settings
- Pull-to-refresh for transaction updates

5.3 Transaction Entry Screen

Desktop Layout:

- Modal dialog with full form
- Two-column layout for form fields
- Auto-suggest dropdowns for payee and category
- Save and continue option

Mobile Layout:

- Full-screen form
- Large touch-friendly inputs
- Step-by-step form progression
- Quick templates for common transactions

5.4 Payee Management Screen

Desktop Layout:

- Searchable list with inline editing
- Bulk actions toolbar
- Usage statistics column
- Quick add payee form

Mobile Layout:

- List view with swipe actions
- Search bar at top
- Floating action button for new payee
- Simple form for adding/editing

5.5 Category Management Screen

Desktop Layout:

- Hierarchical tree view with drag-and-drop
- Category details panel
- Usage statistics and color coding
- Bulk management tools

Mobile Layout:

- Collapsible category list
- Simple hierarchy indication
- Edit categories in separate screen
- Color-coded category indicators

6. Testing Requirements

6.1 Unit Testing Tasks

- Test all API endpoints with various inputs
- Test database operations and constraints
- Test business logic validation
- Test error handling scenarios
- Test authentication and authorization

6.2 Integration Testing Tasks

- Test complete user workflows
- Test cross-device data synchronization
- Test responsive design on various screen sizes
- Test API integration with frontend
- Test third-party service integrations

6.3 User Acceptance Testing Tasks

- Test complete user journey from registration to daily usage
- Test accessibility features
- Test performance under load
- Test security measures
- Test data migration and backup/restore

This Low Level Design document provides comprehensive guidance for implementing a complete personal finance management system with all necessary user stories, technical tasks, and specifications for both frontend and backend development.