ASSET-X (A Mini Bank) Document

Project Overview

Asset-X is a Mini Bank application that provides a set of banking services such as user registration, account management, loan application, transaction handling, and balance tracking. It is designed to allow users to interact with their bank accounts and perform various banking operations, including deposits, withdrawals, and transfers.

Key Features:

1. **User Registration & Authentication:**
    - Users can sign up, log in, and update personal information.
    - Authentication is performed using secure methods (e.g., JWT tokens).
2. **Account Management:**
    - Users can view their accounts and check balances.
    - Supports multiple account types (e.g., savings, checking).
3. **Transactions:**
    - Users can deposit money into their accounts.

- Withdraw funds and perform transfers between accounts.

4. **Loans:**
   - Users can apply for loans, view loan details, and track repayment schedules.
5. **Card Management:**
   - Users can manage debit/credit cards issued by Asset-X.
6. **Audit Logs:**
   - Track user activities (e.g., login, withdrawal) for security and monitoring purposes.
7. **Interest Rates:**
   - Users can view the current interest rates for their savings accounts.

Technology Stack

- Backend: Node.js, Express.js
- Database: MongoDB
- Authentication: JWT (JSON Web Tokens)
- Libraries/Modules:
  - mongoose: ODM for MongoDB

- bcryptjs: For password hashing
- jsonwebtoken: For token-based authentication
- express: For routing and middleware
- body-parser: For parsing incoming request bodies

Database Schema

1. User Document

Stores personal details and login information for each user.

json

CopyEdit

```
{
  "_id": ObjectId("unique_user_id"),
  "firstName": "John",
  "lastName": "Doe",
  "email": "john.doe@example.com",
  "passwordHash": "hashed_password_here",
  "phone": "+1234567890",
  "address": {
    "street": "123 Main St",
    "city": "Anytown",
```

```
    "state": "CA",
    "zip": "12345"
  },
  "dateOfBirth": "1985-06-15",
  "dateJoined": ISODate("2025-03-10T00:00:00Z"),
  "status": "active",
  "lastLogin": ISODate("2025-03-09T15:30:00Z")
}
```

2. Account Document

Represents a bank account belonging to a user.

json

CopyEdit

```
{
  "_id": ObjectId("unique_account_id"),
  "userId": ObjectId("unique_user_id"),
  "accountNumber": "1234567890",
  "accountType": "savings",
  "balance": 1000.50,
  "currency": "USD",
  "status": "active",
```

```
  "dateCreated": ISODate("2025-03-01T00:00:00Z")
}
```

3. Transaction Document

Stores details of each transaction performed by a user.

json

CopyEdit

```
{
  "_id": ObjectId("unique_transaction_id"),
  "accountId": ObjectId("unique_account_id"),
  "transactionType": "deposit",
  "amount": 200.00,
  "currency": "USD",
  "transactionDate": ISODate("2025-03-10T15:00:00Z"),
  "transactionStatus": "completed",
  "description": "Deposit from paycheck",
  "balanceAfterTransaction": 1200.50
}
```

4. Loan Document

Stores details of loans applied by users, including repayment schedules.

json

CopyEdit

```
{
  "_id": ObjectId("unique_loan_id"),
  "userId": ObjectId("unique_user_id"),
  "loanAmount": 5000.00,
  "interestRate": 5.5,
  "loanType": "personal",
  "loanStatus": "active",
  "startDate": ISODate("2025-03-01T00:00:00Z"),
  "endDate": ISODate("2026-03-01T00:00:00Z"),
  "repaymentSchedule": [
    {
      "dueDate": ISODate("2025-04-01T00:00:00Z"),
      "amountDue": 450.00,
      "status": "paid"
    }
  ]
}
```

API Endpoints

1. User Authentication

- POST /api/register: Register a new user
  - Request Body: { "firstName": "John", "lastName": "Doe", "email": "john.doe@example.com", "password": "password123" }
  - Response: 200 OK if registration is successful.
- POST /api/login: User login and JWT token generation
  - Request Body: { "email": "john.doe@example.com", "password": "password123" }
  - Response: 200 OK with JWT token on successful login.

2. Account Management

- GET /api/accounts: Retrieve all accounts of the logged-in user
  - Response: List of accounts with their balances and details.
- POST /api/accounts: Create a new bank account for the user
  - Request Body: { "accountType": "savings" }
  - Response: 201 Created with account details.

3. Transactions

- POST /api/transactions: Create a new transaction (deposit, withdrawal, or transfer)
    - Request Body: { "accountId": "unique_account_id", "transactionType": "deposit", "amount": 200.00 }
    - Response: 200 OK with transaction details.
- GET /api/transactions/:accountId: Get all transactions for a specific account
    - Response: List of transactions for the given account.

4. Loans

- POST /api/loans: Apply for a loan
    - Request Body: { "loanAmount": 5000.00, "loanType": "personal" }
    - Response: 200 OK with loan details.
- GET /api/loans/:userId: Get all loans for a specific user
    - Response: List of loans with repayment schedule.

5. Card Management

- POST /api/cards: Issue a new debit/credit card for the user
    - Request Body: { "cardType": "debit" }
    - Response: 201 Created with card details.

6. Interest Rates

- GET /api/interest-rates: Get the current interest rates for savings accounts
    - Response: List of interest rates for different account types.

Security & Authentication

- JWT Tokens: JWT tokens are used to secure routes and ensure that only authenticated users can access certain API endpoints.
- Password Hashing: Passwords are stored as hashed values using bcryptjs to protect user credentials.
- HTTPS: All API requests should be made over HTTPS to ensure the security of sensitive data.

Project Setup

1. Install Dependencies

bash

CopyEdit

npm install

2. Start the Server

bash

CopyEdit

npm start

3. Environment Variables

Create a .env file and add the following variables:

env

CopyEdit

DB_URI=mongodb://localhost:27017/asset-x

JWT_SECRET=your_jwt_secret_key

Future Enhancements

- Add email verification during registration.
- Implement two-factor authentication (2FA).
- Expand loan types (e.g., home loans, auto loans).
- Add advanced reporting (e.g., transaction history, account statements).