Bank Service Management

# Introduction

The Bank Service system is designed to manage customers, accounts, transactions, and other banking services. It provides CRUD operations for customers and accounts, supports transactions such as deposit, withdrawal, fund transfer, and balance inquiry. The system ensures security with authentication and authorization mechanisms, while maintaining transaction history for audits and compliance. This documentation is intended for developers, testers, and bank staff.

# System Overview

The system is modular and consists of the following modules:

- Customer Management  
- Account Management  
- Transactions  
- Balance Inquiry  
- Loans Management

# Features

• Customer Management: CRUD operations for customer records  
• Account Management: Open, update, close accounts  
• Transactions: Deposit, Withdraw, Fund Transfer  
• Balance Inquiry: Get current account balance  
• Security: Authentication, Authorization, Password rules  
•MongoDB Collections

# Recommended Project Structure

Bank Services Management

│──database.py # MongoDB client connection

│── README.md

│── requirements.txt

│── main.py

│── model.py

│── route.py

│── utility.py

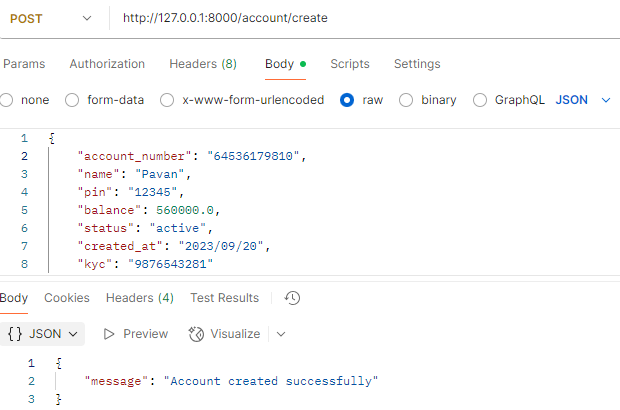
# System Requirements

• Python 3.12 or above  
• FastAPI Framework  
• MongoDB / JSON file for data storage  
• Postman for API testing  
• Uvicorn as ASGI server

# Error Handling

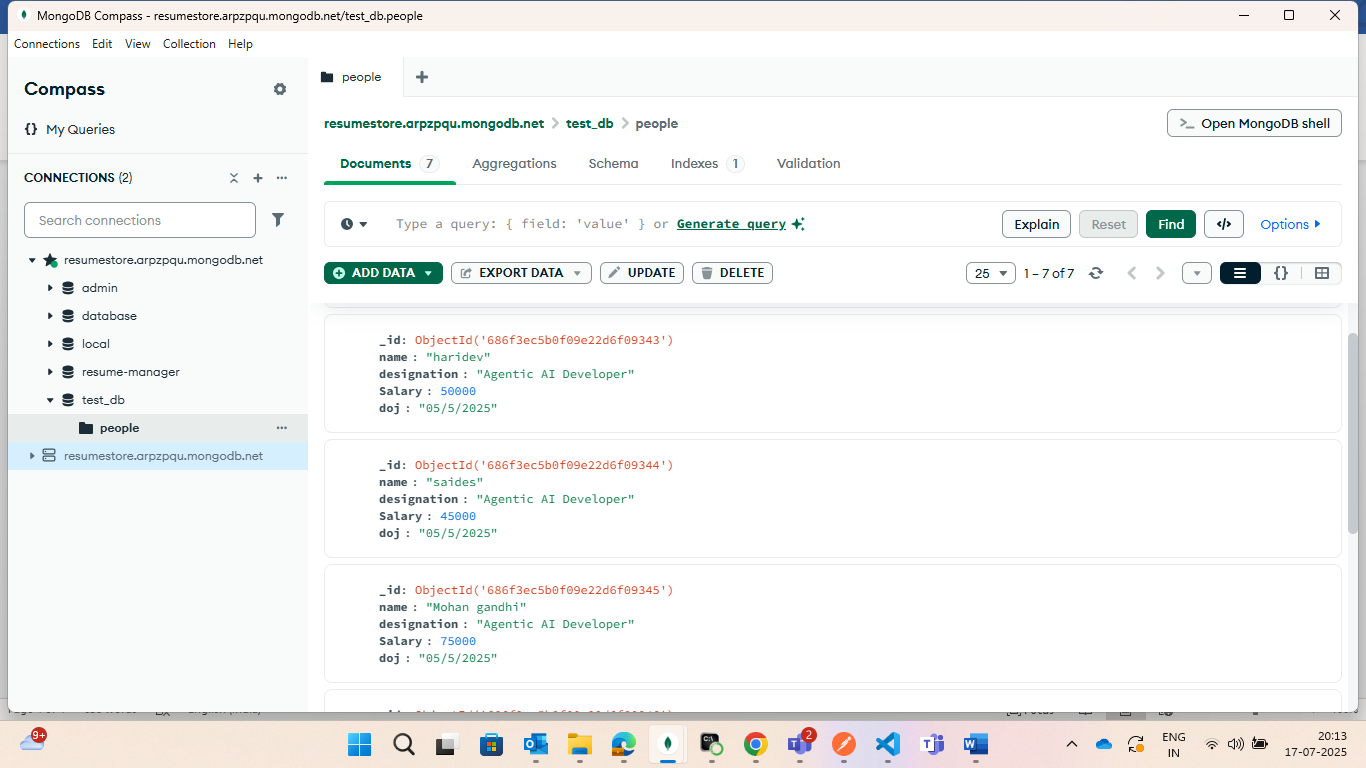
• 400 Bad Request – Invalid input  
• 401 Unauthorized – Login required  
• 404 Not Found – Account/Customer not found  
• 500 Internal Server Error – Unexpected issues

# Postman API Testing

• In Postman, first create a collection and environment with variables like {{base\_url}} and {{token}}.

Test the **login API** to get a JWT token, then use it to call customer, account, and transaction APIs (deposit, withdraw, transfer, balance).

# MongoDB



*MongoDB Comepass:*

•Data Bases = Bank Service Management

*Collections:*

• Accounts Collection  
• Loans Collection  
• Transaction

After testing APIs in Postman, we can authomatically see the data uploaded into MongoDB Collections

## Usage of the Project (Customer’s View)

This **Bank Services API** is designed to let customers manage their banking activities digitally.  
From the **customer side**, it provides the following benefits:

### 1. Account Management

* Create a **new account** with KYC details.
* **Update KYC** (Aadhaar, PAN, Address).
* **Change PIN** securely.
* **Close account** when not needed.
* **Check account balance** anytime.

Example: A customer opens an account online, later updates their address through KYC, and checks their balance via mobile banking.

### 2. Transactions

* **Deposit** money into the account.
* **Withdraw** cash securely with PIN validation.
* **Transfer funds** to another customer’s account.
* **View transaction history** for tracking spending.

Example: A customer transfers ₹5000 to a friend’s account or withdraws cash while ensuring all activity is logged.

### 3. Loan Services

* **Apply for a loan** (personal, home, etc.).
* **View loan details** (amount, interest, tenure, due balance).
* **Repay loan installments** easily.

Example: A customer takes a ₹1,00,000 loan, sees the remaining balance, and pays EMI via API.

### 4. Transparency & Security

* All activities (deposit, withdrawal, loan repayment) are **stored in MongoDB** → ensuring **audit trails**.
* PIN-based validation keeps accounts **secure**.
* Customers can always **track balances & dues**.

## Final Value to Customer

✔️ Customers can perform **banking from anywhere** (web/app) without visiting a branch.  
✔️ Provides **convenience, security, and transparency**.  
✔️ Ensures customers have **control over their accounts, transactions, and loans** digitally.