**Functional Requirements (FR) — Banking System**

*What the system must do.*

**FR-1 Customer Registration**

* Users can create an account by providing **full name, email, phone, username, password**.
* System validates uniqueness of **username/email** and password policy (min length, complexity).
* System sends **verification code** (email/SMS) and activates the profile after verification.
* **Acceptance:** Given valid, unique details → account created, verification required; invalid/duplicate → clear error.

**FR-2 User Authentication (MUST)**

* Customers and Bank Admins can **log in** with username/password.
* System verifies credentials, **hashes** passwords, checks **account status (active/locked)**, and creates a **session/token**.
* **Acceptance:** Valid & active → dashboard shown; invalid or inactive → error; 5 failed attempts → account locked.

**FR-3 Account Management (MUST)**

* Admin can **open** a bank account for a customer: **Savings, Investment, Cheque**.
* Admin can **close** an account (only when **balance = 0** and not locked in disputes).
* Customer can **view all accounts** and **balances**.
* **Acceptance:** New account receives unique **accountNo** and correct type; closed accounts are non-transactable and archived.

**FR-4 Deposit Funds (MUST)**

* Customer can deposit **positive amount** into a selected account.
* System validates **account exists**, **amount > 0**, and records a **Transaction(DEPOSIT)**; **updatedBalance** is shown.
* **Acceptance:** After deposit, **balance increases by amount** and a transaction record is created with timestamp.

**FR-5 Withdraw Funds (MUST)**

* Customer can withdraw a **positive amount** from Savings/Cheque if **sufficient funds** (or within **overdraftLimit** for Cheque).
* System records **Transaction(WITHDRAWAL)** and shows updated balance.
* **Acceptance:** Funds are deducted respecting rules; insufficient funds → error + no transaction created.

**FR-6 Transfer Funds (MUST)**

* Customer can transfer funds **between own accounts** and to **other customers**; optionally to **external bank** via Payment Network.
* System validates **source/destination**, **amount > 0**, **available funds**, creates **paired transactions** (TRANSFER\_OUT / TRANSFER\_IN).
* **Acceptance:** Balances updated on both accounts; external transfers receive a network **ack**.

**FR-7 Transaction History / Statements (SHOULD)**

* Customer can view **paginated** list of transactions per account with **filters** (date range, type).
* System can generate a **monthly statement** (PDF/CSV).
* **Acceptance:** List matches recorded ledger; statement totals reconcile with account balance.

**FR-8 Apply Monthly Interest**

* On **month end** (or admin trigger), system calculates and credits interest for **InterestBearing** accounts with **balance > 0**.
* Records **Transaction**and updates balances.
* **Acceptance:** Interest = **balance × monthlyRate** by account type; audit trail stored.

**FR-9 Profile Management**

* Customer can update **email/phone** (re-verification), and **change password**.
* **Acceptance:** Changes persisted; re-verification required for contact changes.

**FR-10 Audit & Admin Management**

* Admin can **manage customers** (activate/lock), **manage accounts**, and **adjust interest rates** (effective from next cycle).
* **Acceptance:** All admin actions leave an **audit log**

**Non-Functional Requirements (NFR)**

*(How the system should behave.)*

**Security**

* **Password storage:** salted **hash** (e.g., bcrypt/Argon2).
* **Auth:** session or signed token; **idle timeout** 15–30 min.
* **Access control:** role-based (Customer, BankAdmin).
* **Input validation & sanitization** for all forms; prevent injection/XSS.
* **Audit logging:** all monetary operations and admin actions.
* **Lockout:** after 5 failed logins; unlock by admin or cooldown.
* **Transport security:** HTTPS (even if simulated).

**Performance**

* **Auth and balance views < 1s** under normal load.
* **Deposits/withdrawals/transfers < 2s** including persistence.
* **Interest batch** completes within assignment dataset (≤ a few thousand accounts) in **< 60s**.

**Reliability & Availability**

* **Atomic transactions** per operation (all-or-nothing).
* **Data integrity**: no negative balances except allowed overdraft.
* **Backups** (even if simulated export) before monthly interest run.

**Usability**

* Simple, mobile-friendly forms; clear errors near fields.
* Consistent labels and currency formatting; confirmation for destructive actions (close account).

**Maintainability**

* **Layered architecture** (View → Controller/Service → DAO/Repository → Model).
* Unit tests for **BankService**; clear logging; configuration for interest rates.

**Scalability (future)**

* DAO can be swapped for RDBMS; index on accountNo, customerId, and timestamp for transaction queries.

**Compliance (educational scope)**

* No real PII sharing; follow least-privilege access; show consent for storing contact info.

**Mock Interview Record (Appendix)**

*(Simulated Q&A with the “client” = instructor; concise, credible transcript.)*

**Interviewer:** What are the core goals of this banking system?  
**Client:** Allow customers to view balances and perform deposits, withdrawals, and transfers safely. Admins should manage customers/accounts and monthly interest.

**Interviewer:** Which account types are required?  
**Client:** Savings and Investment earn monthly interest (different rates). Cheque supports payments and an overdraft limit.

**Interviewer:** How should authentication work?  
**Client:** Username/password with basic lockout. Store passwords securely and require verification on registration.

**Interviewer:** Any rules on transactions?  
**Client:** Amount must be positive. Withdrawals need enough funds; Cheque can use an overdraft. Transfers must validate both accounts; external transfers should show an acknowledgment.

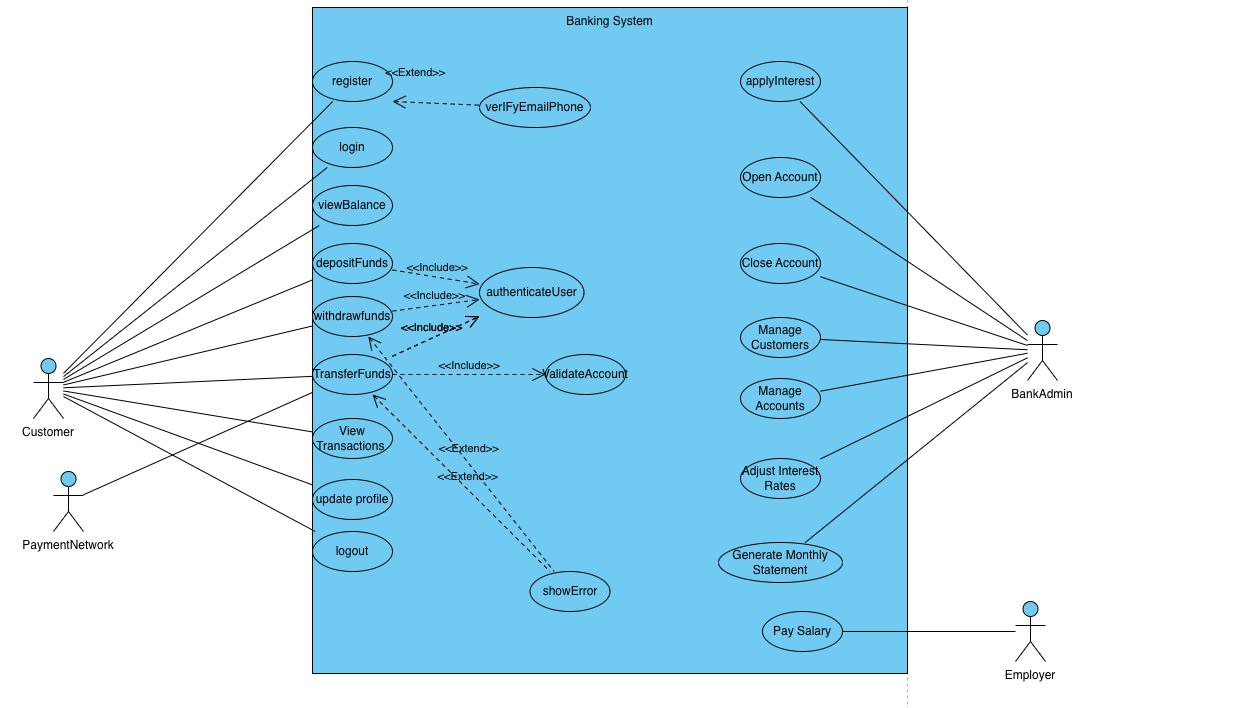
**Interviewer:** What about statements?  
**Client:** Customers need a transaction history and monthly downloadable statement.

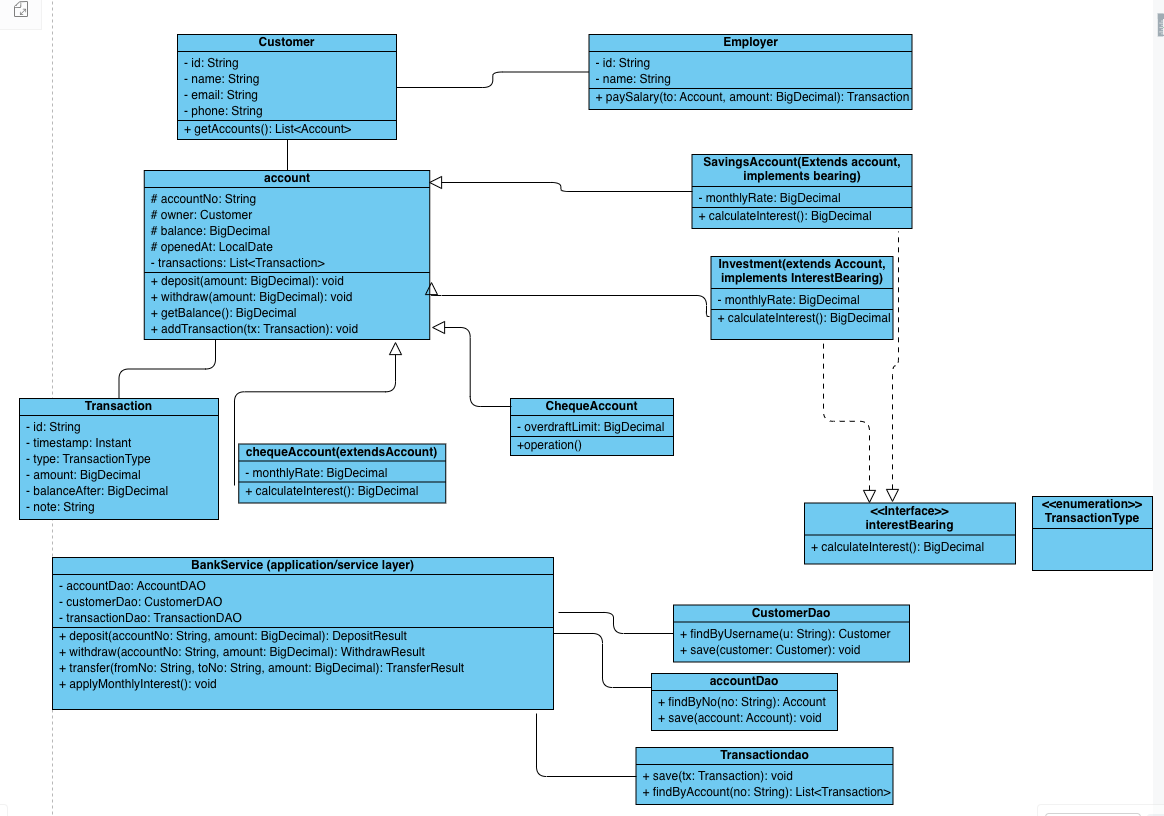
**Interviewer:** Interest calculation details?  
**Client:** Run at month end. Only for interest-bearing accounts with positive balance; record as a transaction.

**Interviewer:** Non-functional expectations?  
**Client:** Secure by default, clear UI, operations responsive within a couple of seconds, and all monetary actions audited.

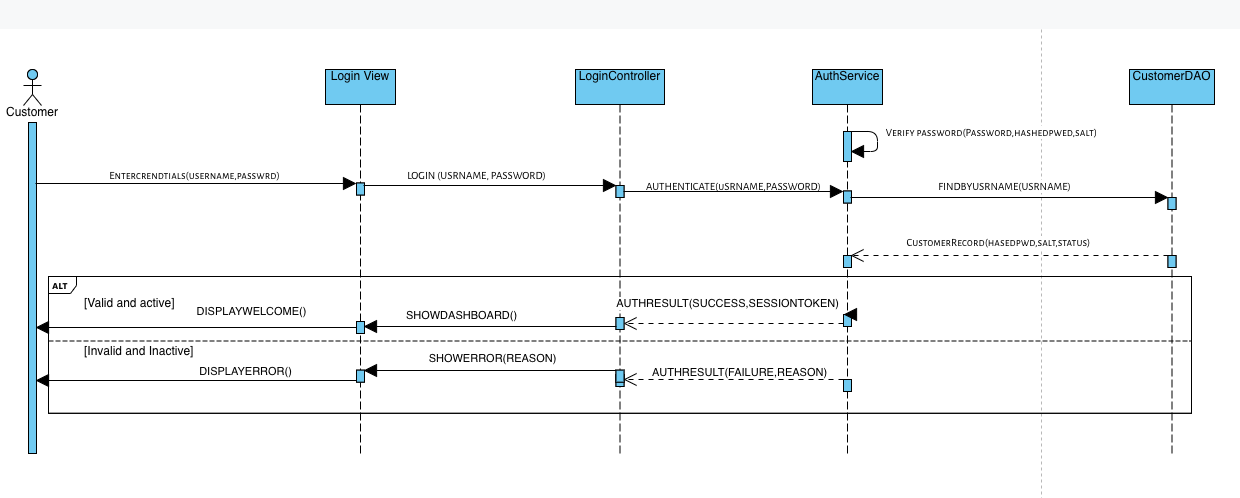
**Interviewer:** Who manages rates and accounts?  
**Client:** Bank Admin sets rates, opens/closes accounts, and can lock/unlock customers.

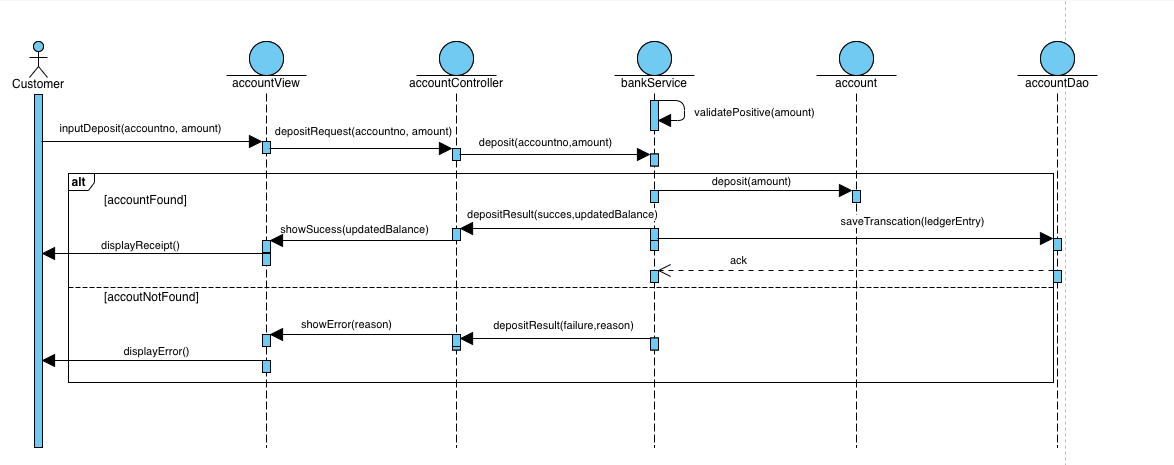
**Interviewer:** Any out-of-scope areas?  
**Client:** Real payment rails and multi-currency are out of scope; we’ll mock the external network.

**Behavioral ModelllingDIAGRAM**

**Class diagram**

Sequence DIAGRAM LOGIN



Sequence diagram deposit  
  
  
  
A diagram of a diagram

Description automatically generated

State Machine