Bank system

Requirements:

1. Manage accounts
    a. customer can create bank account
        i. **Flow** :
            1. Create customer - done [without db]
            2. Create bank unique account
            3. Associate customer with bank account
            4. Return unique account no
    b. Close bank account
        i. **Flow:**
            1. Just remove the bank account, don't delete customer details.
    c. List of bank accounts - [DONE WITH DB]
    d. Get bank account for specific customer
        i. **Flow:**
            1. For provided customer id, we have to give only specified customer bank accounts only.
2. Do Transaction
    a. Credit amount (add transaction amount on current balance of account)
        i. **FLOW**:
            1. Request: toAccountNo, txAmount
            2. Fetch account from db by account no
            3. Add amount to fetched account current balance
            4. Store updated account into db with updated balance
            5. Return account with updated balance
    b. Debit amount (remove transaction amount on current balance of account)
        i. **FLOW:**
            1. Request: fromAccountNo, txAmount
            2. Fetch account from db by account no
            3. Remove amount from fetched account current balance
            4. Validation of current balance check (phase-2)
            5. Store updated account into db with updated balance
            6. Return account with updated balance
3. User
    a. Type of users in bank system
        i. customer
        ii. Bank employee
        iii. Bank manager

    Add required user details into bank account
        iv. **Flow:**
            1. Enter required details and account no

2. Add required details to the account done [without db]
3. Return bank account

How to design a system?

1. requirement analysis (understand requirements, check flow of execution, validation and ask question if you have any thing)we
2. identify entity (college, student, department, faculty, student)
3. identify entity attribute/field (college -> id,name,no, student -> id,name,mobile,email)
4. identify relationship between entity (college 1->M department, department 1->M faculty, department 1->M subject, department 1->M student)
5. identify constraint(any validation if required) (college name should be less than 50 char, student name should be less than 50 char)
6. identify fields for apply indexes (will learn later)
7. apply security (will learn later)

department(did,dname) M<-deparment_facultiy(did,fid,salary)->M faculty (fid,fname)

Start design bank system

Step1:

4. Manage accounts
   a. customer can create bank account
      i. **Flow** :
         1. Create customer
         2. Create bank unique account
         3. Associate customer with bank account
         4. Return unique account no
      ii. **Question**
         1. What if a bank account already exists for customers? Or what if a customer already exists but the account is not created?
            Answer: don't allow customers to create a new account if the account already exists.don't consider the second scenario as of now, we will pick later.
   b. Add required user details into bank account
      i. Flow:
         1. Enter required details and account no
         2. Add required details to the account
         3. Return bank account

        c. Close bank account
            i. **Flow:**
                1. Just remove the bank account, don't delete customer details.
            ii. **Question**:
                1. What do we have to do if the existing balance of the account is more than 0?

                    Answer1: don't worry, bank employees will take care of this scenario. → we don't have to do any code or validation for this scenario.

                    Answer2: We should not allow a bank account if we have more than 0 balance. And suggest customers to withdraw all money first and then close their bank account. → we have to throw an error if balance > 0 while closing the account.
        d. List of bank accounts
        e. Get bank account for specific customer
            i. **Flow:**
                1. Give only specified customer bank account only.
5. Do Transaction
    a. Credit amount (add transaction amount on current balance of account)
    b. Debit amount (remove transaction amount on current balance of account)
        i. **Question**:
            1. What if the current account balance is less than the transaction amount? Ex: current account balance = 250, transaction = 500

                Answer: not allow this transaction and give error to user that "insufficient balance"
6. User
    a. Type of users in bank system
        i. customer
        ii. Bank employee
        iii. Bank manager

Step2:

identify entity

Identified Entity list:

1. User [type: customer, bank_employee, bank_manager]
2. Account
3. Transaction [type: credit, debit]

Step3:

identify entity attribute/field

Identified Entity list:

1. User [type: customer, bank_employee, bank_manager]
    a. Unique user id
    b. Type
    c. Name
        i. First name
        ii. Last name
        iii. Middle name
    d. Birth date
    e. Unique pan card
    f. Unique aadhar card
2. Account
    a. unique account no
    b. Account type
    c. User id
    d. Current Balance
    e. Status [open, close]
3. Transaction [type: credit, debit]
    a. Unique Transaction id
    b. Transaction type
    c. From account id
    d. To account id
    e. Date and time
    f. Amount

Step4:

identify relationship between entity

Transaction ←——----- M-1-------------→ Account ←——-----------------1-1 —----------> User

NOTE:

1. Account is not create without user(customer) if customer not exist then throw error.
2. When we do a transaction, we have to add 2 transaction entries on form account and to account.
3. And we also have to change both account current balance
    a. Example:
        i. Customer 1 does one transaction of 200 rupees from account 101(current balance = 5000) to account 201(current balance = 2000). Then we have two entities of transaction as bellow
            1. Account 101, amount 200 rupees, type = debit
            2. Account 201, amount 200 rupees, type = credit

           Then we have to change both account current balance as per transaction

1. Account 101, current bal(5000) - tx amount(200) = 4800
2. Account 201, current bal(2000) + tx amount(200) = 2200

Step5:

identify constraint(any validation if required)

→ as per flow and question answer and notes we have to add validation and constraint.
If required then add constraints here.

Example:
Requirement said that firstname should be 50 char long only, then we have to validate and add constraint on firstname field.

Codding start:

1. Create class from entity list
2. Create fields inside classes
3. Create relationship fields inside classes(if 1-1 or 1-M or M-1 relationship) or create new class(if M-M relationship)