**Q:Bank Account Management System - Design a class to manage bank accounts, including account balance, withdrawals, and deposits.**

**Approach to solve:**

1.Understand the problem statement clearly and find out the all the end points need to be created .

2.Figure out the proper datasturctures to manipulate the data in the api

3.Figure out the database design with proper list of tables and appropriate constraints to solve the problem.

**Pseudocode:**

**CREATE\_ACCOUNT (HTTP POST method):**

Step 1: Extract the required data from the JSON payload.

Step 2: If the data format is not correct, display a message saying "Invalid request format".

Step 3: Create a database connection using the set\_connection() method. If it is created successfully, insert the extracted data into the bank1 table and save changes. If any exception is raised, roll back the transaction and display a message saying "Failed to create the account".

Step 4: Close all the connections and display a message saying "Account created successfully".

SHOW\_LIST (HTTP GET method):

Step 1: Create a database connection using the set\_connection() method. If it is created successfully, execute the SELECT query to fetch all the data from the bank1 table.

Step 2: Fetch the data using fetchall() method.

Step 3: Close all the connections and return the fetched data as a string along with the status code 200.

**WITHDRAWAL (HTTP PUT method):**

Step 1: Create a database connection using the set\_connection() method. If it is created successfully, extract the srno and withdraw\_amount data from the JSON payload.

Step 2: If the data format is not correct or srno or amount are missing, display a message saying "Missing required field(s)".

Step 3: Execute the SELECT query to fetch the balance amount of the account from the bank1 table using srno.

Step 4: If the account is not found, display a message saying "Account not found".

Step 5: If the balance is less than the amount to be withdrawn, display a message saying "Insufficient balance".

Step 6: Update the balance by subtracting the amount to be withdrawn from the fetched balance.

Step 7: Execute the UPDATE query to update the balance of the account with the new balance amount.

Step 8: Close all the connections and return the updated amount and new balance as a JSON response along with the status code 200.

**DEPOSIT (HTTP PUT method):**

Step 1: Create a database connection using the set\_connection() method. If it is created successfully, extract the srno and deposit\_amount data from the JSON payload.

Step 2: If the data format is not correct or srno or amount are missing, display a message saying "Missing required field(s)".

Step 3: Execute the SELECT query to fetch the balance amount of the account from the bank1 table using srno.

Step 4: If the account is not found, display a message saying "Account not found".

Step 5: Update the balance by adding the amount to be deposited to the fetched balance. Step 6: Execute the UPDATE query to update the balance of the account with the new balance amount.

Step 7: Close all the connections and return the deposited amount and new balance as a JSON response along with the status code 200.

**DELETE\_ACCOUNT (HTTP DELETE method):**

Step 1: Create a database connection using the set\_connection() method. If it is created successfully, execute the DELETE query to delete the record with the provided srno from the bank1 table.

Step 2: Close all the connections and display a message saying "Record deleted successfully".