#include <iostream>

#include <vector>

#include <unordered\_map>

#include <iomanip>

using namespace std;

class BankAccount {

private:

string username;

string password;

double balance;

vector<string> transactions;

static unordered\_map<string, BankAccount> accounts;

public:

BankAccount() : balance(0.0) {}

void registerAccount() {

cout << "Enter username: ";

cin >> username;

if (accounts.find(username) != accounts.end()) {

cout << "Username already exists. Try another.\n";

return;

}

cout << "Enter password: ";

cin >> password;

balance = 0.0;

accounts[username] = \*this;

cout << "Account registered successfully!\n";

}

bool login() {

string user, pass;

cout << "Enter username: ";

cin >> user;

cout << "Enter password: ";

cin >> pass;

if (accounts.find(user) != accounts.end() && accounts[user].password == pass) {

\*this = accounts[user];

cout << "Login successful!\n";

return true;

}

cout << "Invalid username or password.\n";

return false;

}

void deposit() {

double amount;

cout << "Enter deposit amount: ";

cin >> amount;

if (amount > 0) {

balance += amount;

transactions.push\_back("Deposit: " + to\_string(amount));

accounts[username] = \*this;

cout << "Deposit successful! New balance: " << fixed << setprecision(2) << balance << "\n";

} else {

cout << "Invalid amount.\n";

}

}

void withdraw() {

double amount;

cout << "Enter withdrawal amount: ";

cin >> amount;

if (amount > 0 && amount <= balance) {

balance -= amount;

transactions.push\_back("Withdrawal: " + to\_string(amount));

accounts[username] = \*this;

cout << "Withdrawal successful! New balance: " << fixed << setprecision(2) << balance << "\n";

} else {

cout << "Invalid or insufficient funds.\n";

}

}

void checkBalance() {

cout << "Current balance: " << fixed << setprecision(2) << balance << "\n";

}

void viewTransactionHistory() {

if (transactions.empty()) {

cout << "No transaction history available.\n";

return;

}

cout << "\nTransaction History:\n";

for (const string& t : transactions) {

cout << t << "\n";

}

}

};

unordered\_map<string, BankAccount> BankAccount::accounts;

int main() {

BankAccount account;

int choice;

bool loggedIn = false;

while (true) {

cout << "\nBanking System\n";

cout << "1. Register\n2. Login\n3. Deposit\n4. Withdraw\n5. Check Balance\n6. View Transaction History\n7. Exit\n";

cout << "Choose an option: ";

cin >> choice;

switch (choice) {

case 1:

account.registerAccount();

break;

case 2:

loggedIn = account.login();

break;

case 3:

if (loggedIn) account.deposit();

else cout << "Please log in first.\n";

break;

case 4:

if (loggedIn) account.withdraw();

else cout << "Please log in first.\n";

break;

case 5:

if (loggedIn) account.checkBalance();

else cout << "Please log in first.\n";

break;

case 6:

if (loggedIn) account.viewTransactionHistory();

else cout << "Please log in first.\n";

break;

case 7:

cout << "Exiting system.\n";

return 0;

default:

cout << "Invalid choice, please try again.\n";

}

}

return 0;

}