AZKA AREEJ

L1F23BSCS0229

BANK MANAGEMENT SYSTEM

#include <iostream>

#include <fstream>

using namespace std;

struct Account {

int accountNumber;

int accountpassword;

char name[50];

double balance;

};

Account\* accounts = nullptr;

int accountCount = 0;

void loadAccounts() {

ifstream fin("accounts.txt");

if (!fin.is\_open()) return;

Account acc;

while (fin >> acc.accountNumber >> acc.accountpassword >> acc.name >> acc.balance) {

Account\* newAccounts = new Account[accountCount + 1];

for (int i = 0; i < accountCount; ++i) {

newAccounts[i] = accounts[i];

}

newAccounts[accountCount] = acc;

delete[] accounts;

accounts = newAccounts;

accountCount++;

}

fin.close();

}

void saveAccounts() {

ofstream fout("accounts.txt");

for (int i = 0; i < accountCount; ++i) {

fout << accounts[i].accountNumber << " " << accounts[i].accountpassword << " " << accounts[i].name << " " << accounts[i].balance << endl;

}

fout.close();

}

void logTransaction(int accountNumber, int accountpassword, const char\* type, double amount) {

ofstream fout2("transaction.txt", ios::app);

fout2 << accountNumber << " " << accountpassword << " " << type << " " << amount << endl;

fout2.close();

}

int generateAccountNumber() {

int AccountNumber = 0;

if (accountCount > 0) {

int lastAccountNumber = accounts[accountCount - 1].accountNumber;

AccountNumber = lastAccountNumber + 1;

}

else {

AccountNumber = 1;

}

return AccountNumber;

}

void createNewAccount() {

cout << "---------------------------------------------" << endl;

Account newAccount;

cout << "Enter name: ";

cin.ignore();

cin.getline(newAccount.name, 50);

cout << "Enter initial deposit: ";

cin >> newAccount.balance;

cout << "Enter account password: ";

cin >> newAccount.accountpassword;

newAccount.accountNumber = generateAccountNumber();

Account\* newAccounts = new Account[accountCount + 1];

for (int i = 0; i < accountCount; ++i) {

newAccounts[i] = accounts[i];

}

newAccounts[accountCount] = newAccount;

delete[] accounts;

accounts = newAccounts;

accountCount++;

saveAccounts();

cout << "Account created successfully" << endl;

cout << "Account Number : " << newAccount.accountNumber << endl;

cout << "---------------------------------------------" << endl;

}

void depositMoney() {

cout << "---------------------------------------------" << endl;

int pass;

int accnum;

double amount;

cout << "Enter account number: ";

cin >> accnum;

cout << "Enter account password: ";

cin >> pass;

bool accountFound = false;

for (int i = 0; i < accountCount; ++i) {

if (accounts[i].accountNumber == accnum && accounts[i].accountpassword == pass) {

accountFound = true;

cout << "Enter amount to deposit: ";

cin >> amount;

accounts[i].balance += amount;

saveAccounts();

logTransaction(accnum, pass, "Deposit", amount);

cout << "Deposit successful. New balance: " << accounts[i].balance << endl;

cout << "---------------------------------------------" << endl;

return;

}

}

if (!accountFound) {

cout << "Account not found or invalid password" << endl;

cout << "---------------------------------------------" << endl;

}

}

void withdrawMoney() {

cout << "---------------------------------------------" << endl;

int pass;

int accnum;

double amount;

cout << "Enter account number: ";

cin >> accnum;

cout << "Enter account password: ";

cin >> pass;

bool accountFound = false;

for (int i = 0; i < accountCount; ++i) {

if (accounts[i].accountNumber == accnum && accounts[i].accountpassword == pass) {

accountFound = true;

cout << "Enter amount to withdraw: ";

cin >> amount;

if (accounts[i].balance >= amount) {

accounts[i].balance -= amount;

saveAccounts();

logTransaction(accnum, pass, "Withdrawal", amount);

cout << "Withdrawal successful. New balance: " << accounts[i].balance << endl;

cout << "---------------------------------------------" << endl;

}

else {

cout << "Insufficient balance" << endl;

cout << "---------------------------------------------" << endl;

}

return;

}

}

if (!accountFound) {

cout << "Account not found or invalid password" << endl;

cout << "---------------------------------------------" << endl;

}

}

void checkAccountBalance() {

cout << "---------------------------------------------" << endl;

int accNum;

cout << "Enter account number: ";

cin >> accNum;

for (int i = 0; i < accountCount; ++i) {

if (accounts[i].accountNumber == accNum) {

cout << "Current balance: " << accounts[i].balance << endl;

cout << "---------------------------------------------" << endl;

return;

}

}

cout << "Account not found" << endl;

cout << "---------------------------------------------" << endl;

}

void viewAccountDetails() {

cout << "---------------------------------------------" << endl;

int accNum;

cout << "Enter account number: ";

cin >> accNum;

for (int i = 0; i < accountCount; ++i) {

if (accounts[i].accountNumber == accNum) {

cout << "Account Number: " << accounts[i].accountNumber << endl;

cout << "Name: " << accounts[i].name << endl;

cout << "Balance: " << accounts[i].balance << endl;

cout << "Acoount Password: " << accounts[i].accountpassword << endl;

cout << "---------------------------------------------" << endl;

return;

}

}

cout << "Account not found" << endl;

cout << "---------------------------------------------" << endl;

}

void viewAllAccounts() {

if (accountCount > 0) {

cout << "---------------------------------------------" << endl;

for (int i = 0; i < accountCount; ++i) {

cout << "Account Number: " << accounts[i].accountNumber << endl;

cout << "Name: " << accounts[i].name << endl;

cout << "Balance: " << accounts[i].balance << endl;

cout << "Acoount Password: " << accounts[i].accountpassword << endl;

cout << "---------------------------------------------" << endl;

}

}

else {

cout << "Oops there are no accounts to display" << endl;

}

}

void searchForAccount() {

cout << "---------------------------------------------" << endl;

int accNum;

cout << "Enter account number: ";

cin >> accNum;

for (int i = 0; i < accountCount; ++i) {

if (accounts[i].accountNumber == accNum) {

cout << "Account Number: " << accounts[i].accountNumber << endl;

cout << "Name: " << accounts[i].name << endl;

cout << "Balance: " << accounts[i].balance << endl;

cout << "Acoount Password: " << accounts[i].accountpassword << endl;

cout << "---------------------------------------------" << endl;

return;

}

}

cout << "Account not found" << endl;

cout << "---------------------------------------------" << endl;

}

void deleteAccount() {

cout << "---------------------------------------------" << endl;

int accNum;

cout << "Enter account number: ";

cin >> accNum;

for (int i = 0; i < accountCount; ++i) {

if (accounts[i].accountNumber == accNum) {

for (int j = i; j < accountCount - 1; ++j) {

accounts[j] = accounts[j + 1];

}

accountCount--;

saveAccounts();

cout << "Account deleted successfully" << endl;

cout << "---------------------------------------------" << endl;

return;

}

}

cout << "Account not found" << endl;

cout << "---------------------------------------------" << endl;

}

void modifyAccountDetails() {

cout << "---------------------------------------------" << endl;

int accNum;

cout << "Enter account number: ";

cin >> accNum;

for (int i = 0; i < accountCount; ++i) {

if (accounts[i].accountNumber == accNum) {

cout << "Enter new name: ";

cin.ignore();

cin.getline(accounts[i].name, 50);

saveAccounts();

cout << "Account details updated successfully" << endl;

cout << "---------------------------------------------" << endl;

return;

}

}

cout << "Account not found" << endl;

cout << "---------------------------------------------" << endl;

}

void adminMenu() {

while (true) {

cout << "\n--- Administration Menu ---\n";

cout << "1. View All Accounts\n";

cout << "2. Search for an Account\n";

cout << "3. Delete an Account\n";

cout << "4. Modify Account Details\n";

cout << "0. Return to Main Menu\n";

cout << "Enter your choice: ";

int choice;

cin >> choice;

switch (choice) {

case 1:

viewAllAccounts();

break;

case 2:

searchForAccount();

break;

case 3:

deleteAccount();

break;

case 4:

modifyAccountDetails();

break;

case 0:

return;

default:

cout << "Invalid choice. Please try again.\n";

}

}

}

void userMenu() {

while (true) {

cout << "\n--- User Menu ---\n";

cout << "1. Create New Account\n";

cout << "2. Deposit Money\n";

cout << "3. Withdraw Money\n";

cout << "4. Check Account Balance\n";

cout << "5. View Account Details\n";

cout << "0. Return to Main Menu\n";

cout << "Enter your choice: ";

int choice;

cin >> choice;

switch (choice) {

case 1:

createNewAccount();

break;

case 2:

depositMoney();

break;

case 3:

withdrawMoney();

break;

case 4:

checkAccountBalance();

break;

case 5:

viewAccountDetails();

break;

case 0:

return;

default:

cout << "Invalid choice. Please try again.\n";

}

}

}

int main() {

loadAccounts();

while (true) {

cout << "\n--- Bank Management System ---\n";

cout << "1. User\n";

cout << "2. Administrator\n";

cout << "0. Exit\n";

cout << "Enter your choice: ";

int choice;

cin >> choice;

switch (choice) {

case 1:

userMenu();

break;

case 2:

adminMenu();

break;

case 0:

delete[] accounts;

cout << "Exiting program...\n";

return 0;

default:

cout << "Invalid choice. Please try again.\n";

}

}

return 0;

}