Name: Shahd Allam Mohamed

```cpp
#include <iostream>
#include <vector>
#include <string>
using namespace std;

class Transaction {
public:
    string type;
    double amount;
    string description;

    Transaction(string t, double a, string d) : type(t), amount(a), description(d) {}
};

class Account {
public:
    int accountNumber;
    double balance;
    vector<Transaction> history;

    Account(int accNum) : accountNumber(accNum), balance(0.0) {}

    void deposit(double amount) {
        balance += amount;
        history.push_back(Transaction("Deposit", amount, "Deposited to account"));
        cout << "Deposit successful. New Balance: " << balance << endl;
    }

    bool withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            history.push_back(Transaction("Withdrawal", amount, "Withdrawn from account"));
            cout << "Withdrawal successful. New Balance: " << balance << endl;
            return true;
        } else {
            cout << "Insufficient balance." << endl;
            return false;
        }
    }

    bool transfer(Account &toAccount, double amount) {
        if (balance >= amount) {
            balance -= amount;
            toAccount.balance += amount;
            history.push_back(Transaction("Transfer Out", amount, "Transferred to account " + to_string(toAccount.accountNumber)));
            toAccount.history.push_back(Transaction("Transfer In", amount, "Received from account " + to_string(accountNumber)));
            cout << "Transfer successful. New Balance: " << balance << endl;
            return true;
        } else {
            cout << "Insufficient balance." << endl;
            return false;
        }
    }

    void showAccountInfo() {
        cout << "Account Number: " << accountNumber << ", Balance: " << balance << endl;
        cout << "Transaction History: " << endl;
        for (auto &t : history) {
            cout << t.type << " - Amount: " << t.amount << " - " << t.description << endl;
        }
    }
};

class Customer {
public:
    string name;
    int customerID;
    vector<Account> accounts;

    Customer(string n, int id) : name(n), customerID(id) {}

    void addAccount(Account account) {
        accounts.push_back(account);
    }

    Account* getAccount(int accNum) {
        for (auto &acc : accounts) {
            if (acc.accountNumber == accNum) return &acc;
        }
        return nullptr;
    }

    void showCustomerInfo() {
        cout << "Customer ID: " << customerID << ", Name: " << name << endl;
        for (auto &acc : accounts) {
            acc.showAccountInfo();
            cout << "-----------------------------" << endl;
        }
    }
};
```

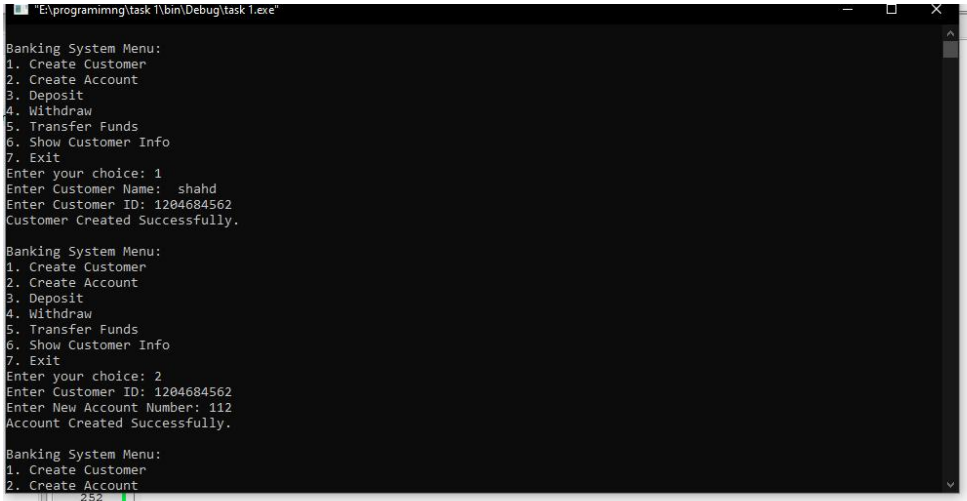
```cpp
int main() {
    vector<Customer> customers;
    int choice;

    while (true) {
        cout << "\nBanking System Menu:\n";
        cout << "1. Create Customer\n";
        cout << "2. Create Account\n";
        cout << "3. Deposit\n";
        cout << "4. Withdraw\n";
        cout << "5. Transfer Funds\n";
        cout << "6. Show Customer Info\n";
        cout << "7. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;

        if (choice == 1) {
            string name;
            int id;
            cout << "Enter Customer Name: ";
            cin >> name;
            cout << "Enter Customer ID: ";
            cin >> id;
            customers.push_back(Customer(name, id));
            cout << "Customer Created Successfully." << endl;
        } else if (choice == 2) {
            int custID, accNum;
            cout << "Enter Customer ID: ";
            cin >> custID;

            int custID, accNum;
            cout << "Enter Customer ID: ";
            cin >> custID;

            Customer *customer = nullptr;
            for (auto &c : customers) {
                if (c.customerID == custID) {
                    customer = &c;
                    break;
                }
            }

            if (customer) {
                cout << "Enter New Account Number: ";
                cin >> accNum;
                customer->addAccount(Account(accNum));
                cout << "Account Created Successfully." << endl;
            } else {
                cout << "Customer Not Found." << endl;
            }
        } else if (choice == 3) {
            int custID, accNum;
            double amount;
            cout << "Enter Customer ID: ";
            cin >> custID;

            Customer *customer = nullptr;
            for (auto &c : customers) {
                if (c.customerID == custID) {
                    customer = &c;
                    break;
```

```
                break;
            }
        }

        if (customer) {
            cout << "Enter Account Number: ";
            cin >> accNum;
            Account *acc = customer->getAccount(accNum);
            if (acc) {
                cout << "Enter Deposit Amount: ";
                cin >> amount;
                acc->deposit(amount);
            } else {
                cout << "Account Not Found." << endl;
            }
        } else {
            cout << "Customer Not Found." << endl;
        }
    } else if (choice == 4) {
        int custID, accNum;
        double amount;
        cout << "Enter Customer ID: ";
        cin >> custID;

        Customer *customer = nullptr;
        for (auto &c : customers) {
            if (c.customerID == custID) {
                customer = &c;
                break;
            }
        }

        if (customer) {
            cout << "Enter Account Number: ";
            cin >> accNum;
            Account *acc = customer->getAccount(accNum);
            if (acc) {
                cout << "Enter Withdrawal Amount: ";
                cin >> amount;
                acc->withdraw(amount);
            } else {
                cout << "Account Not Found." << endl;
            }
        } else {
            cout << "Customer Not Found." << endl;
        }
    } else if (choice == 5) {
        int custID, fromAccNum, toAccNum;
        double amount;
        cout << "Enter Customer ID: ";
        cin >> custID;

        Customer *customer = nullptr;
        for (auto &c : customers) {
            if (c.customerID == custID) {
                customer = &c;
                break;
            }
        }

        if (customer) {
```

```
    if (customer) {
        cout << "Enter From Account Number: ";
        cin >> fromAccNum;
        cout << "Enter To Account Number: ";
        cin >> toAccNum;

        Account *fromAcc = customer->getAccount(fromAccNum);
        Account *toAcc = customer->getAccount(toAccNum);

        if (fromAcc && toAcc) {
            cout << "Enter Transfer Amount: ";
            cin >> amount;
            fromAcc->transfer(*toAcc, amount);
        } else {
            cout << "One or both accounts not found." << endl;
        }
    } else {
        cout << "Customer Not Found." << endl;
    }
} else if (choice == 6) {
    int custID;
    cout << "Enter Customer ID: ";
    cin >> custID;

    Customer *customer = nullptr;
    for (auto &c : customers) {
        if (c.customerID == custID) {
            customer = &c;
            break;
        }
    }
```

C/C++ | Windows (CR+LF) | WINDOWS-1252 | Line 153, Col 6, Pos 4695 | Insert | Read/Write | def

```
            Customer *customer = nullptr;
            for (auto &c : customers) {
                if (c.customerID == custID) {
                    customer = &c;
                    break;
                }
            }

            if (customer) {
                customer->showCustomerInfo();
            } else {
                cout << "Customer Not Found." << endl;
            }
        } else if (choice == 7) {
            cout << "Exiting Program." << endl;
            break;
        } else {
            cout << "Invalid Choice." << endl;
        }
    }

    return 0;
}
```




```
252
253         return 0;
```

```
Account Created Successfully.

Banking System Menu:
1. Create Customer
2. Create Account
3. Deposit
4. Withdraw
5. Transfer Funds
6. Show Customer Info
7. Exit
Enter your choice: 7
Exiting Program.

Process returned 0 (0x0)   execution time : 48.703 s
Press any key to continue.
```