

## Documento de Arquitetura de Software

Daniel Alves Cordeiro

Edson Junior Frota Silva

Guilherme Marcorio de Santana

João Victor Dezidério Vilela

Luiggi Giovanucci Faleiro de Freitas Nascimento

### 1. Introdução

Este documento fornece uma visão da arquitetura do *Sistema Bancário Distribuído (DBS)*, onde este foi desenvolvido pelo grupo de alunos do curso de Engenharia de Computação da UFG: Daniel Alves Cordeiro, Edson Júnior Frota da Silva, Guilherme Marcorio de Santana, João Victor Dezidério Vilela e Luiggi Giovanucci Faleiro de Freitas.

Diante disto, o documento possui duas visões distintas da arquitetura do *DBS* com seus diferentes aspectos. Onde o sistema desenvolvido cumpri os requisitos estabelecidos no Projeto Final de Sistemas Distribuídos (documento anexado).

### 2. Representação da Arquitetura

Este documento apresenta a arquitetura das seguintes formas de visualização: visualização lógica e visualização física (implementação). Nota-se que ambas visualizações são em um modelo UML (linguagem de modelagem unificada).

### 3. Objetivos e Restrições da Arquitetura

O *Sistema Bancário Distribuído* possui alguns requisitos que têm um relacionamento significativo com a arquitetura. São eles:

1. A partir do requisito seis, contido no documento anexado, utilizou-se de um servidor master para realização do balanceamento de carga. Além disso, por haver a necessidade de dois servidores, usou-se dois PCs, sendo um servidor para cada.
2. Diante do requisito dois, presente no documento anexado, foi implementado um banco de dados com duas tabelas, conta e movimentação, para realizar o armazenamento dos dados bancários.

### 4. Visualização de Caso de Uso

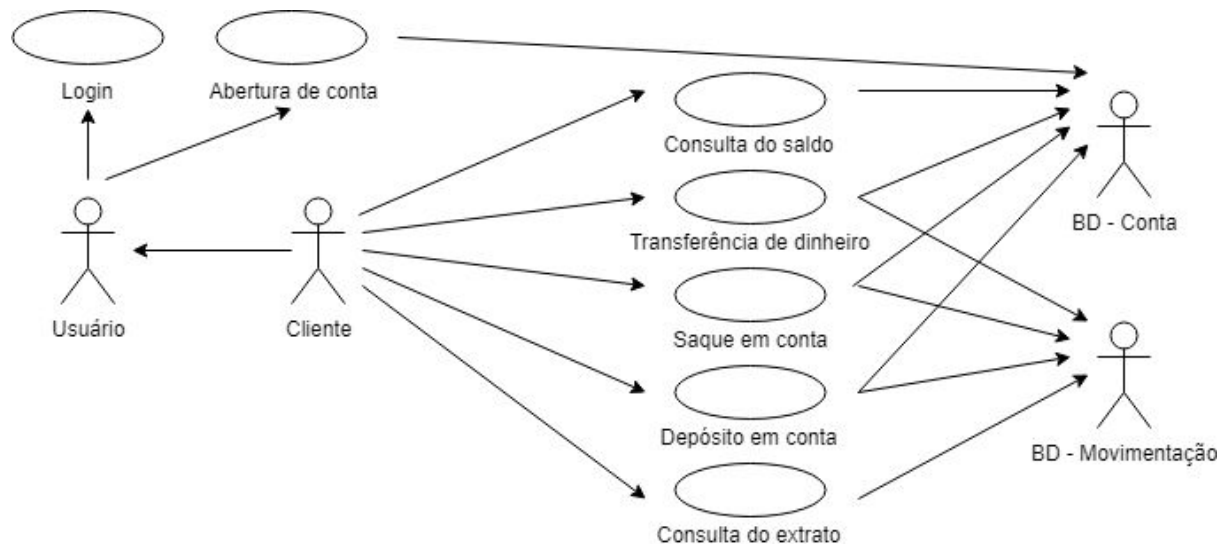
A visualização de Caso de Uso é um método importante para descrição do conjunto de cenários e/ou casos de uso que representam alguma funcionalidade no projeto do *Sistema Bancário Distribuído*. Além disso, este pode contemplar casos de uso e/ou cenários que enfatizam ou ilustram um determinado ponto complicado da arquitetura.

Os casos de uso do *DBS* apresentados abaixo são todos iniciados pelos agentes clientes. Além disso, ocorre interação com o banco de dados, dividido em conta e movimentação.

- Login;
- Abertura de conta;
- Consulta de saldo;
- Consulta de extrato das movimentações financeiras;

- Transferência de dinheiro entre contas do mesmo banco;
- Saque em conta;
- Depósito em conta.

#### 4.1 Casos de Uso Significativos para Arquitetura



##### 4.1.1 Login

Breve descrição: Este caso de uso descreve como um usuário efetua login no Sistema Bancário Distribuído.

##### 4.1.2 Abertura de conta

Breve descrição: Este caso de uso permite que um Cliente possa realizar a abertura de uma conta no banco. Neste caso, é necessário realizar o cadastro de um número, que será sua conta, e de uma senha, para realização do acesso a mesma.

##### 4.1.3 Consulta de saldo

Breve descrição: Este caso de uso permite que um Cliente consulte o saldo de sua conta. Mas para que isso seja realizado é necessário acessar a tabela de conta do Banco de Dados.

##### 4.1.4 Consulta do extrato

Breve descrição: Este caso de uso permite que um Cliente consulte o extrato das movimentações financeiras de sua conta. Assim, é indispensável o acesso a tabela de movimentação presente no Banco de Dados.

##### 4.1.5 Transferência de dinheiro

Breve descrição: Este caso de uso permite que um Cliente realiza a transferência de dinheiro de sua conta para conta de outro Cliente do mesmo banco. Logo, para que tal funcionalidade seja implementada é essencial a consulta e alteração do Banco de Dados.

##### 4.1.6 Saque em conta

Breve descrição: Este caso de uso permite que um Cliente realize o saque de uma quantia de dinheiro presente em sua conta. Nota-se que é preciso o acesso ao Banco de Dados, conta e movimentação.

#### 4.1.7 Depósito em conta

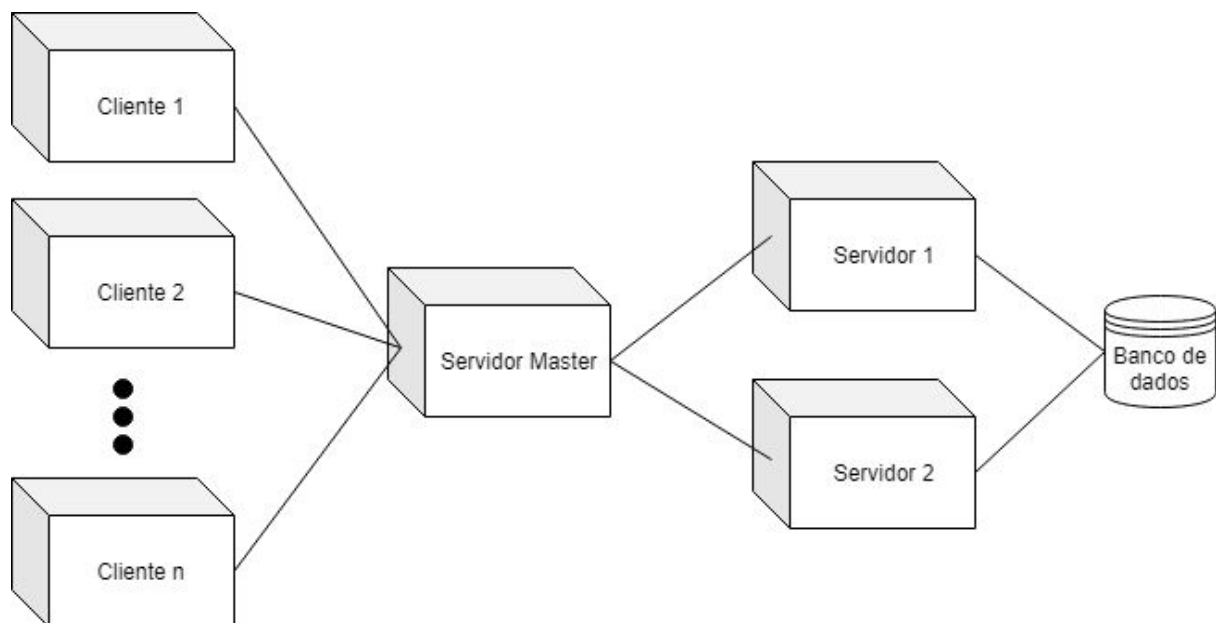
Breve descrição: Este caso de uso permite que um Cliente realize depósito de uma quantia de dinheiro em sua conta, para que isto ocorra é fundamental a alteração de dados presente no Banco de Dados.

### 5. Visualização Lógica

A visualização lógica da arquitetura tem como objetivo definir os componentes utilizados para o desenvolvimento do projeto do *Sistema Bancário Distribuído (DBS)*, fora isso, estabelecer a finalidade destes.

Dados isto, para a implementação do *DBS* será necessário conter os seguintes componente: Banco de Dados, dois Servidores para atender as requisições, um Servidor Master e os Clientes.

#### 5.1 Visão Geral da Arquitetura



##### 5.1.1 Cliente

Breve descrição: o projeto *DBS* conterá pelo menos um Cliente. Este, como já descrito anteriormente, terá que realizar o login no sistema do banco com os dados de sua conta, número e senha, os quais serão checados no Banco de Dados do *DBS*. Ao ser validado o login, o mesmo irá selecionar uma operação a qual será repassada ao Servidor Master.

O sistema em questão, como já especificado, contém as seguintes operações: consulta de extrato, saque em conta, depósito em conta, transferência ou logout.

##### 5.1.2 Servidor Master

Breve descrição: este componente do sistema tem como única função realizar o balanceamento de carga do *Sistema Bancário Distribuído*. Desta forma, este será projetado para ser um mediador das requisições dos Clientes para o Servidor do banco.

### 5.1.3 Servidor 1 e 2

Breve descrição: estes componentes possuem a função principal do projeto DBS, ou seja, eles iram receber as requisições dos clientes e a partir de seus parâmetros invoca o método específico de um determinado objeto, que neste caso é o Banco.

### 5.1.4 Banco de dados

Breve descrição: a partir do requisito dois, o qual determina que o projeto deverá garantir a consistência e armazenamento dos dados, implica a indispensabilidade da utilização de um Banco de Dados. Posto isto, foi definido que este conterà duas tabelas, conta e movimentação, para gerenciar todos os dados a serem utilizados e tratados no projeto.

A tabela da conta, foi projetada para conter os seguintes atributos: nome do cliente, número da conta do cliente, senha da conta e saldo. Enquanto a tabela da movimentação armazena os seguintes dados: identificador da transação, número da conta operada, data da transação, valor, tipo da transação (saque, depósito e transferência), tipo (operação de crédito ou débito) e conta de destino, a qual será utilizada quando for uma transferência.

## 6. Teste

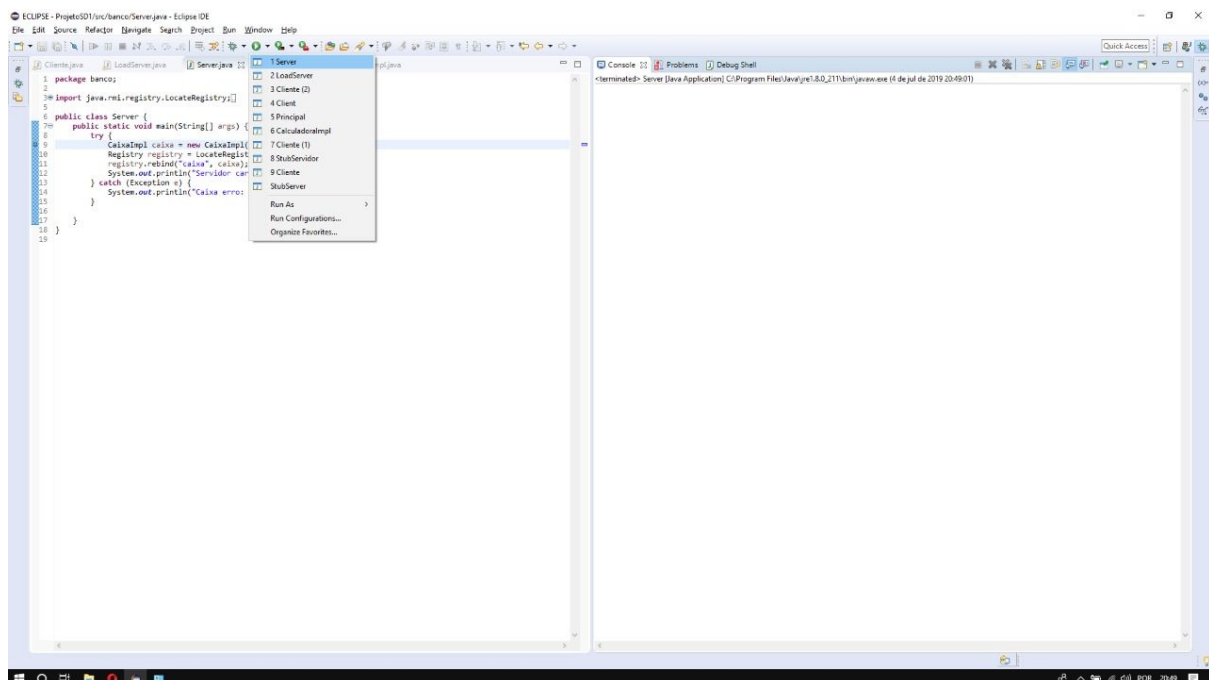
### 6.1. Computadores utilizados

Computador 1: Possui Servidor de balanceamento(LoadServer) e servidor 1;

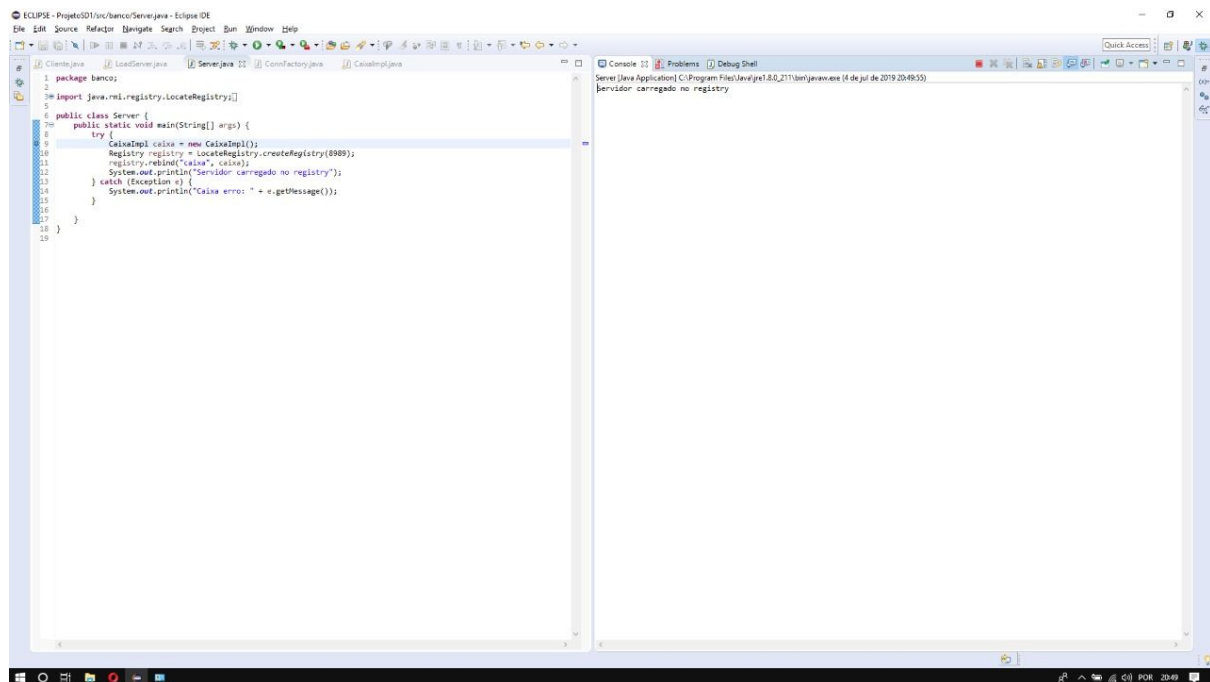
Computador 2: Possui Servidor de Banco de Dados e servidor 2, e cliente;

Computador 3: Possui somente Cliente.

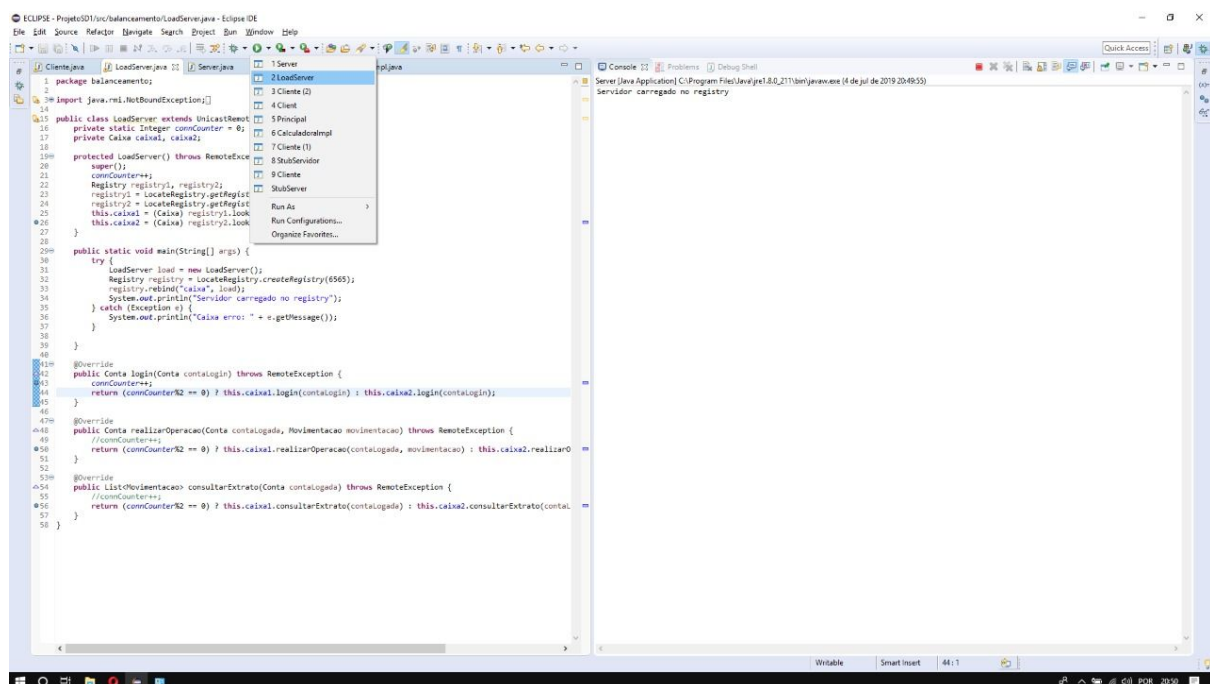
### 6.2. Servidores



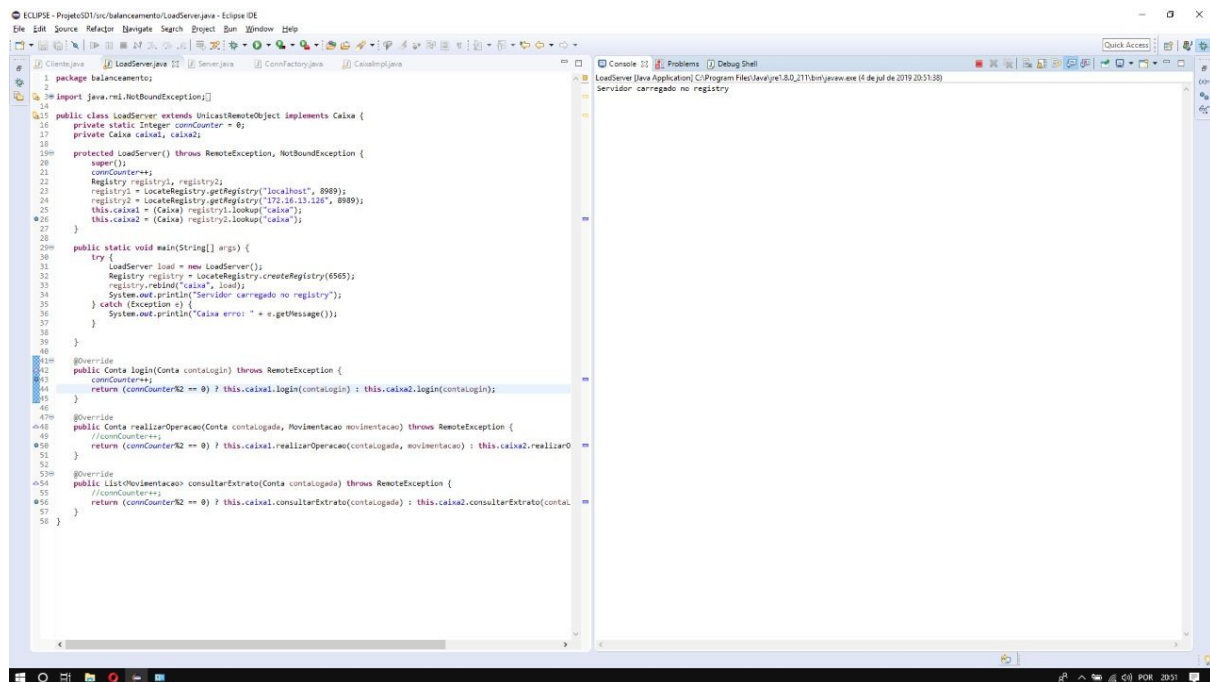
Iniciando servidor computador 1.



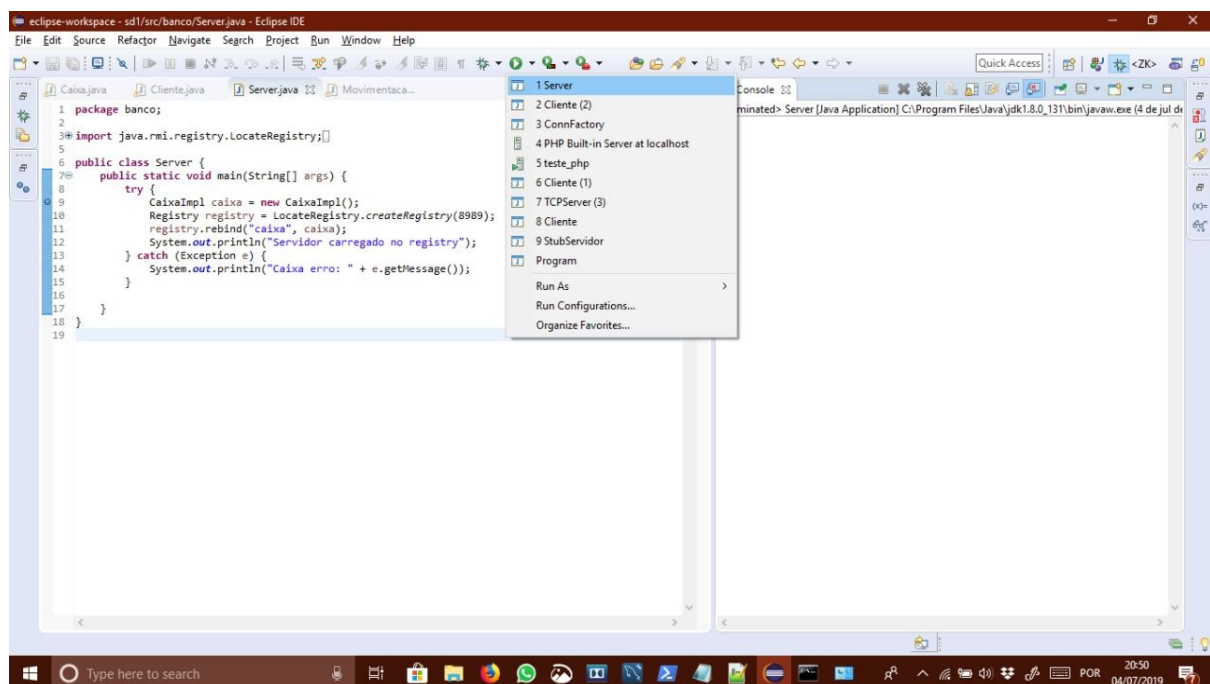
Servidor computador 1 carregado



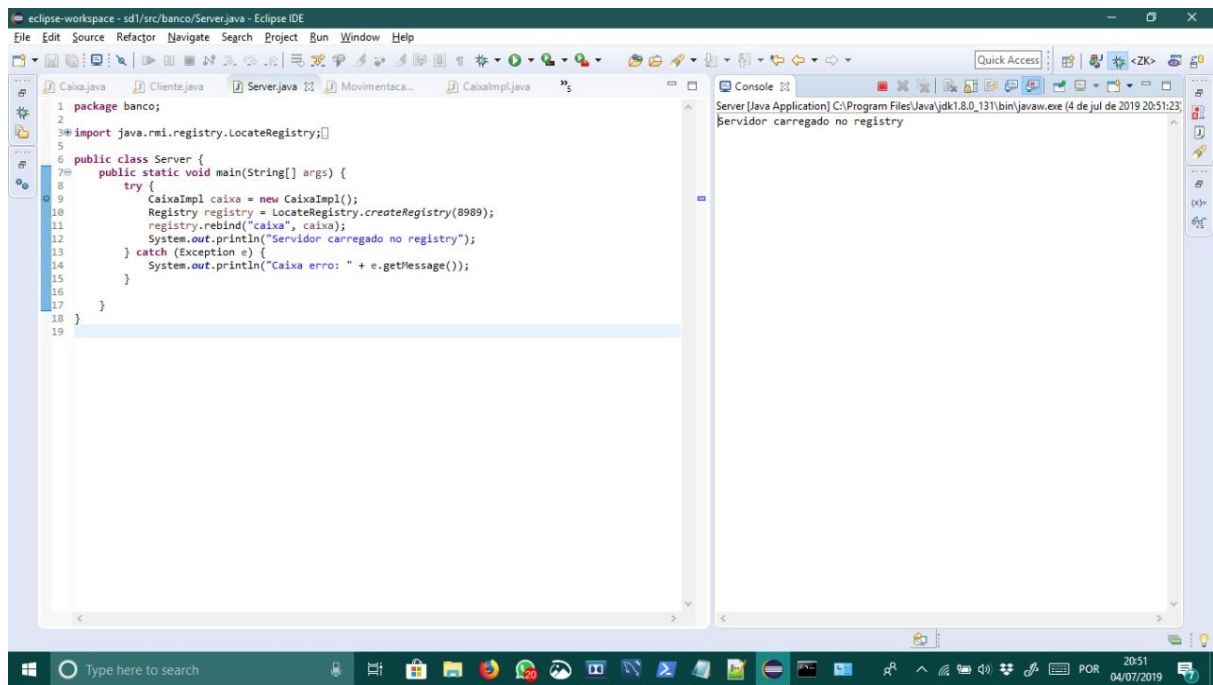
Iniciando LoadServer (Computador 1)



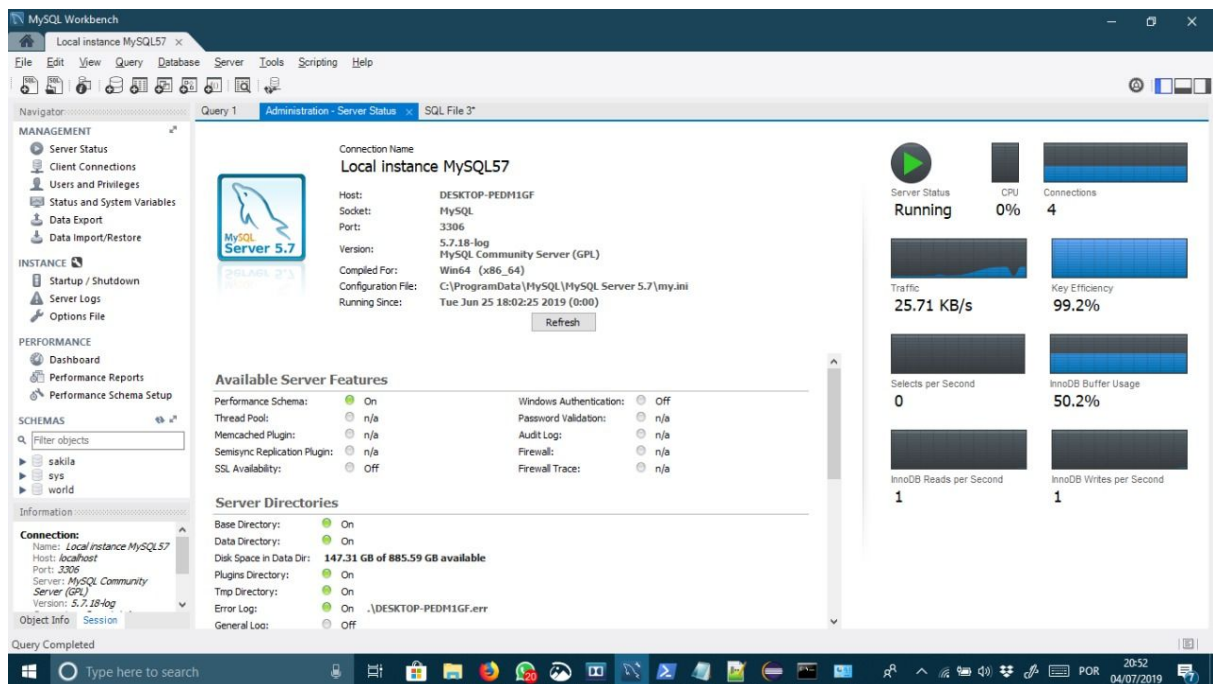
LoadServer carregado(computador 1)



Iniciando servidor(Computador 2)



Servidor computador 2 iniciado.

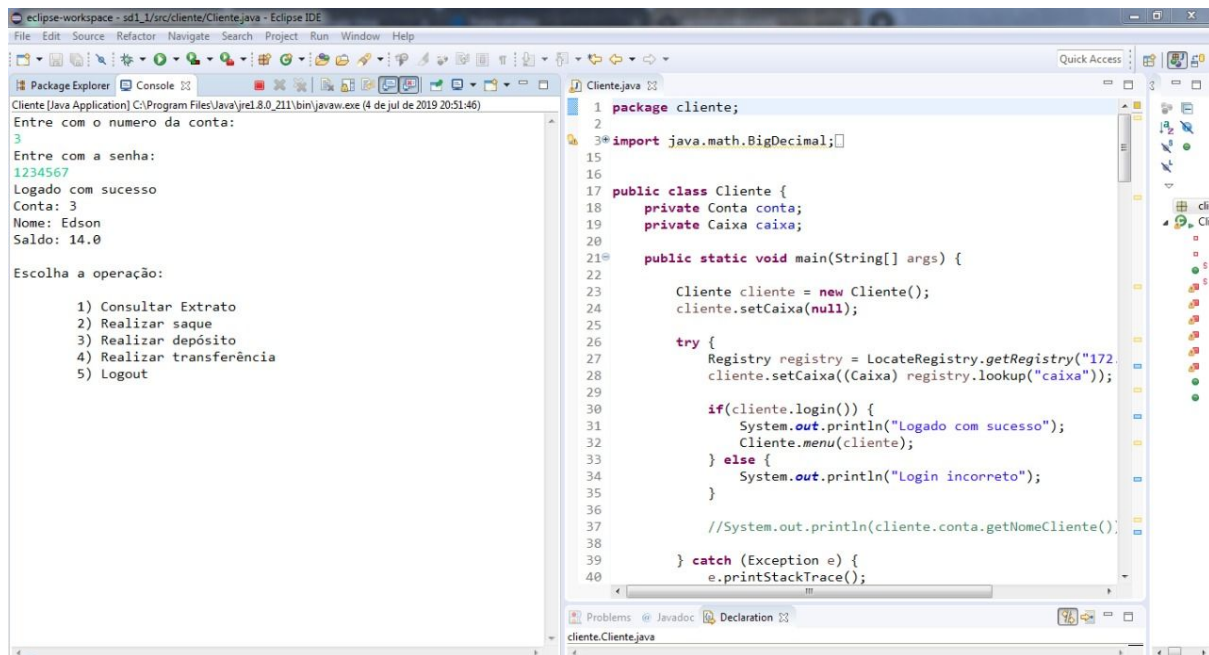


Servidor de banco de dados (computador 2).

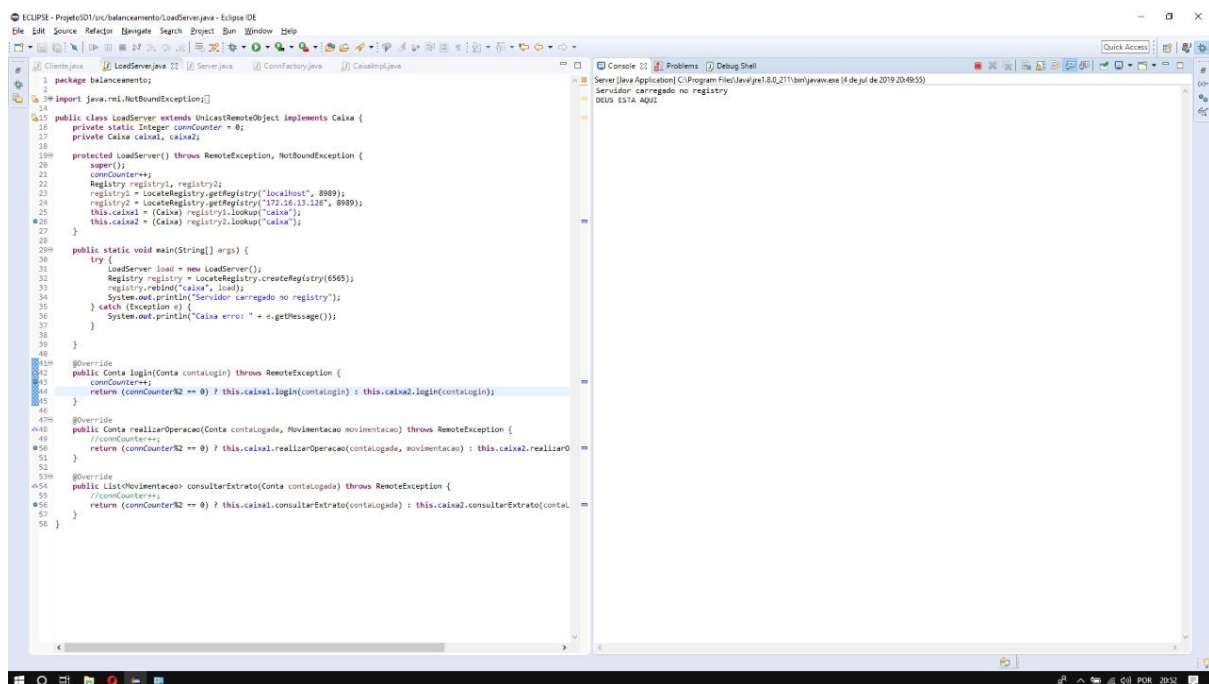
### 6.3. Clientes

Vão realizar as operações de consultar extrato, Saque, Depósito e transferência.

E ainda login e logout.



Login cliente.



Servidor recebeu conexão do cliente logado.



```

1 package cliente;
2
3 import java.math.BigDecimal;
4
5
6
7
8
9
10
11
12
13
14
15
16
17 public class Cliente {
18     private Conta conta;
19     private Caixa caixa;
20
21     public static void main(String[] args) {
22
23         Cliente cliente = new Cliente();
24         cliente.setCaixa(null);
25
26         try {
27             Registry registry = LocateRegistry.getRegistry("172.
28             cliente.setCaixa((Caixa) registry.lookup("caixa"));
29
30             if(cliente.login()) {
31                 System.out.println("Logado com sucesso");
32                 Cliente.menu(cliente);
33             } else {
34                 System.out.println("Login incorreto");
35             }
36
37             //System.out.println(cliente.conta.getNomeCliente());
38
39         } catch (Exception e) {
40             e.printStackTrace();
41         }
42     }
43 }

```

Console Output:

```

<terminated> Cliente [Java Application] C:\Program Files\Java\jdk-8.0_211\bin\javaw.exe (4 de jul de 2019 20:51:46)
Entre com o numero da conta:
3
Entre com a senha:
1234567
Logado com sucesso
Conta: 3
Nome: Edson
Saldo: 14.0

Escolha a operação:

1) Consultar Extrato
2) Realizar saque
3) Realizar depósito
4) Realizar transferência
5) Logout

```

Extrato		
15/04/2019 00:00:00	SAQUE	-100001,00
11/03/2019 00:00:00	SAQUE	-990,00
08/02/2019 01:00:00	SAQUE	-480,00
01/01/2019 01:00:00	DEPOSITO	+100,00
01/01/2019 01:00:00	SAQUE	-120,00
Saldo Final:		14,00

Consulta de extrato.

```

189 ArrayList<Movimentacao> extrato;
190
191 try {
192     extrato = (ArrayList<Movimentacao>) this.caixa.cons
193
194     System.out.println("===== Extrato =====");
195     for (Movimentacao mov : extrato) {
196         String sign = mov.getTipo().equals("c") ? "+" : "-";
197         String valor = String.format("%1s%.2f", sign, mov.getValor());
198         System.out.printf("%15s\t%15s\t\t%15s\n", mov.getData(), mov.getTipo(), valor);
199     }
200
201     System.out.printf("\n\t\t\t\t\tSaldo Final: %10.2f", cliente.getSaldo());
202 } catch (RemoteException e) {
203     // TODO Auto-generated catch block
204     e.printStackTrace();
205 }
206
207
208
209 public Caixa getCaixa() {
210     return caixa;
211 }
212
213 public void setCaixa(Caixa caixa) {
214     this.caixa = caixa;
215 }
216

```

Console Output:

```

Cliente [Java Application] C:\Program Files\Java\jdk-8.0_211\bin\javaw.exe (4 de jul de 2019 20:57:55)
Entre com o numero da conta:
3
Entre com a senha:
1234567
Logado com sucesso
Conta: 3
Nome: Edson
Saldo: 4.0

Escolha a operação:

1) Consultar Extrato
2) Realizar saque
3) Realizar depósito
4) Realizar transferência
5) Logout

```

Insira o valor do saque a ser realizado:

```

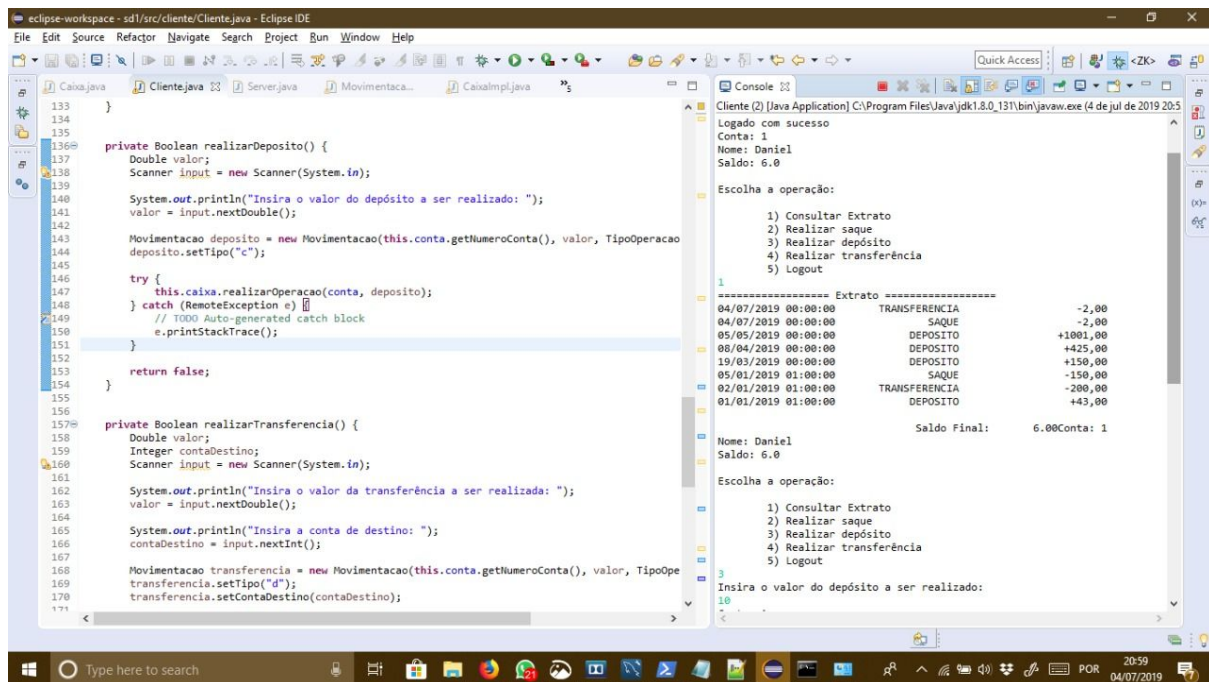
10
Saldo indisponivel
Conta: 3
Nome: Edson
Saldo: 4.0

Escolha a operação:

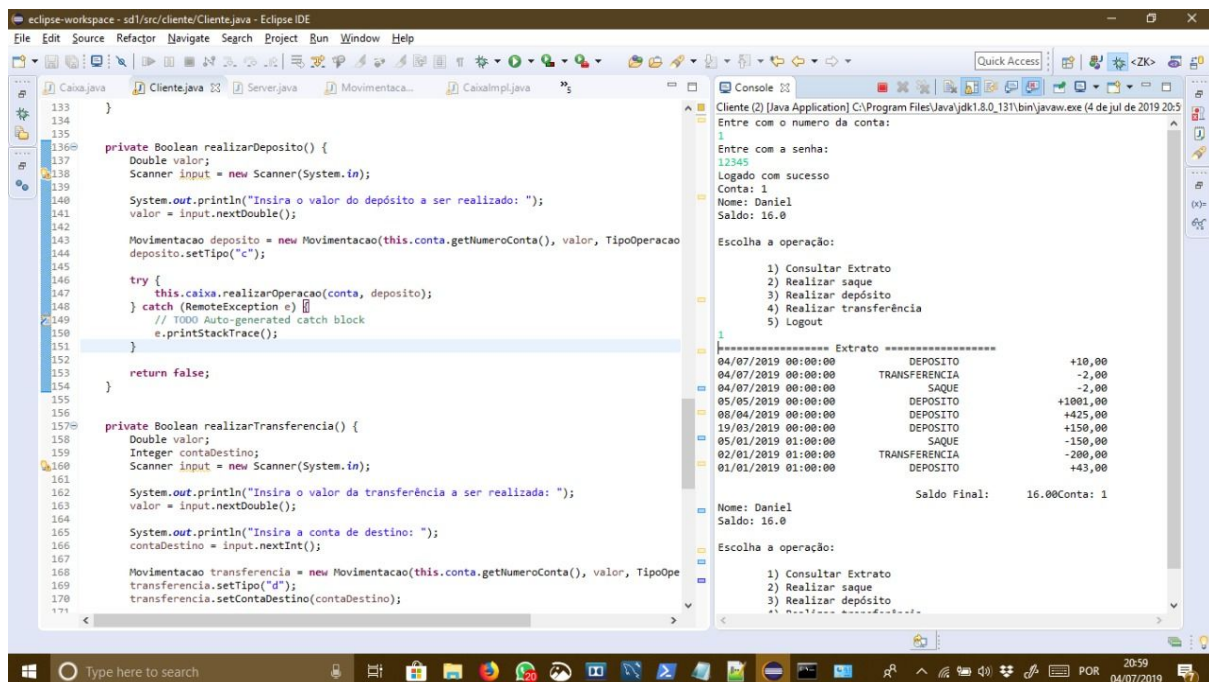
1) Consultar Extrato
2) Realizar saque
3) Realizar depósito
4) Realizar transferência
5) Logout

```

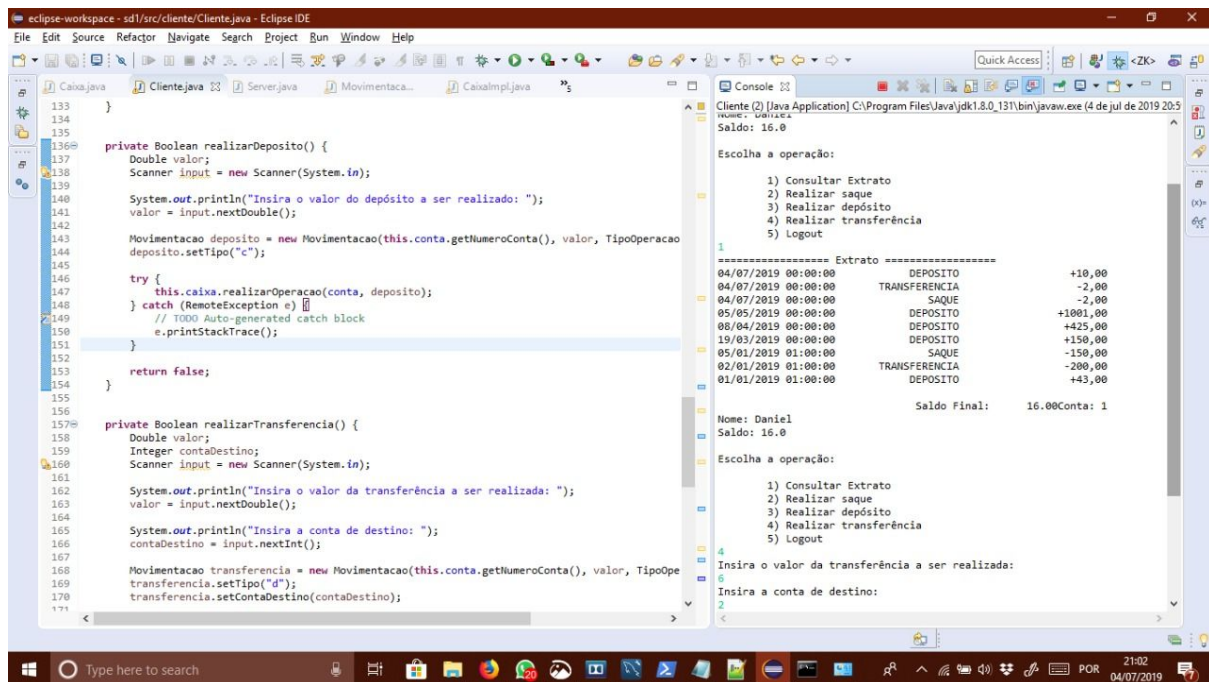
Saque



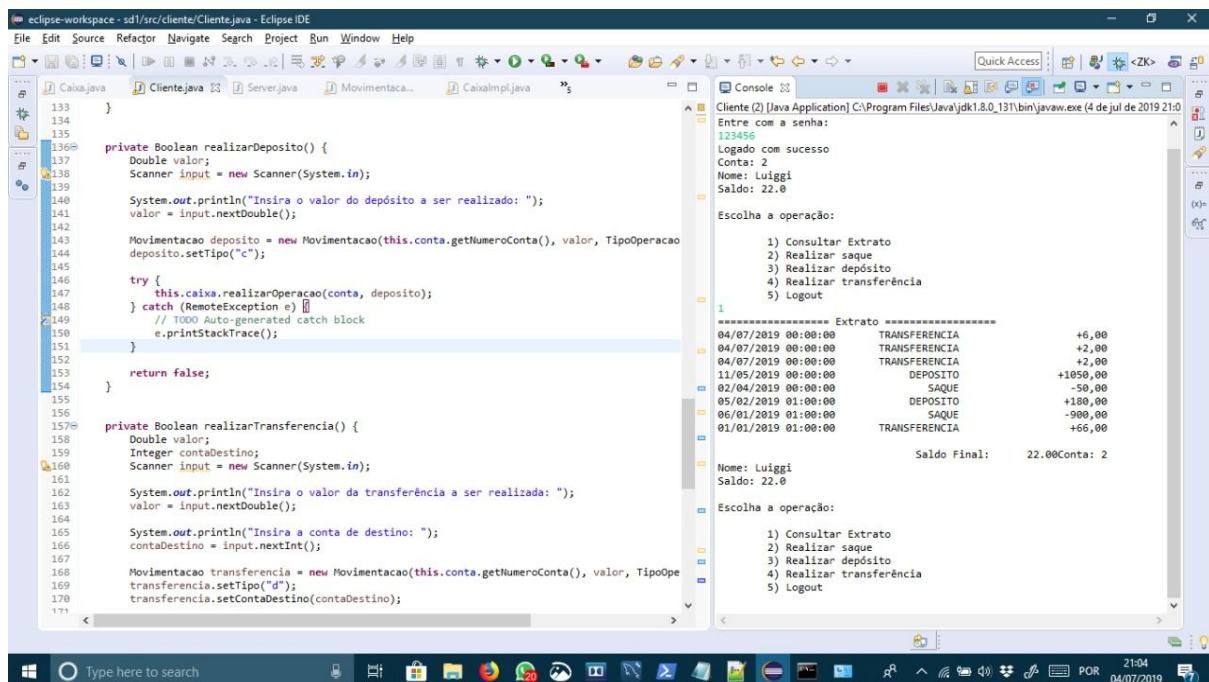
Depósito.



Cliente fazendo consulta de saldo após depósito.



Transferência.



O cliente que recebeu a transferência.