

# **Actividad [#3] - [Funcionalidad a Sistema en Java]**

**[Lenguajes de Programación IV]**

**Ingeniería en Desarrollo de Software**

**Tutor: Juan Miguel Martínez Maldonado**

**Alumno: Alan David López Rojas**

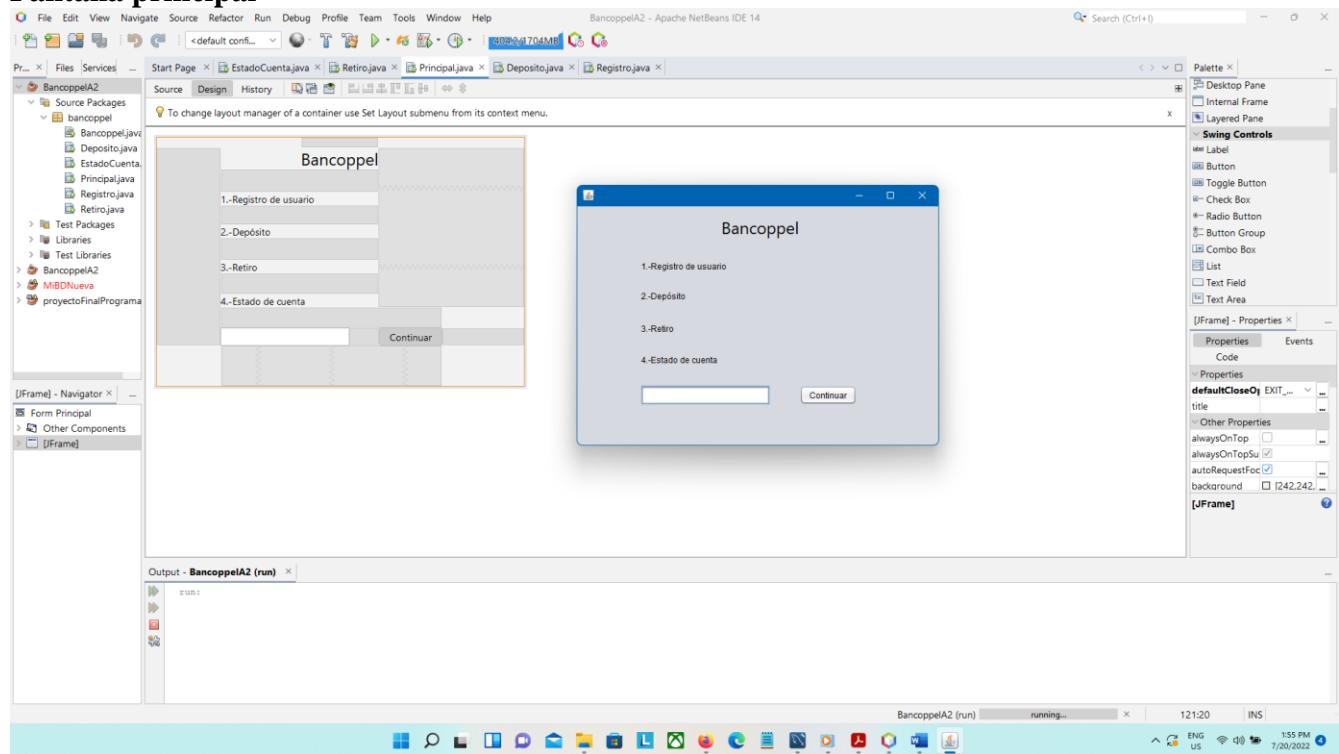
**Fecha: 20/07/2022**

## **Índice**

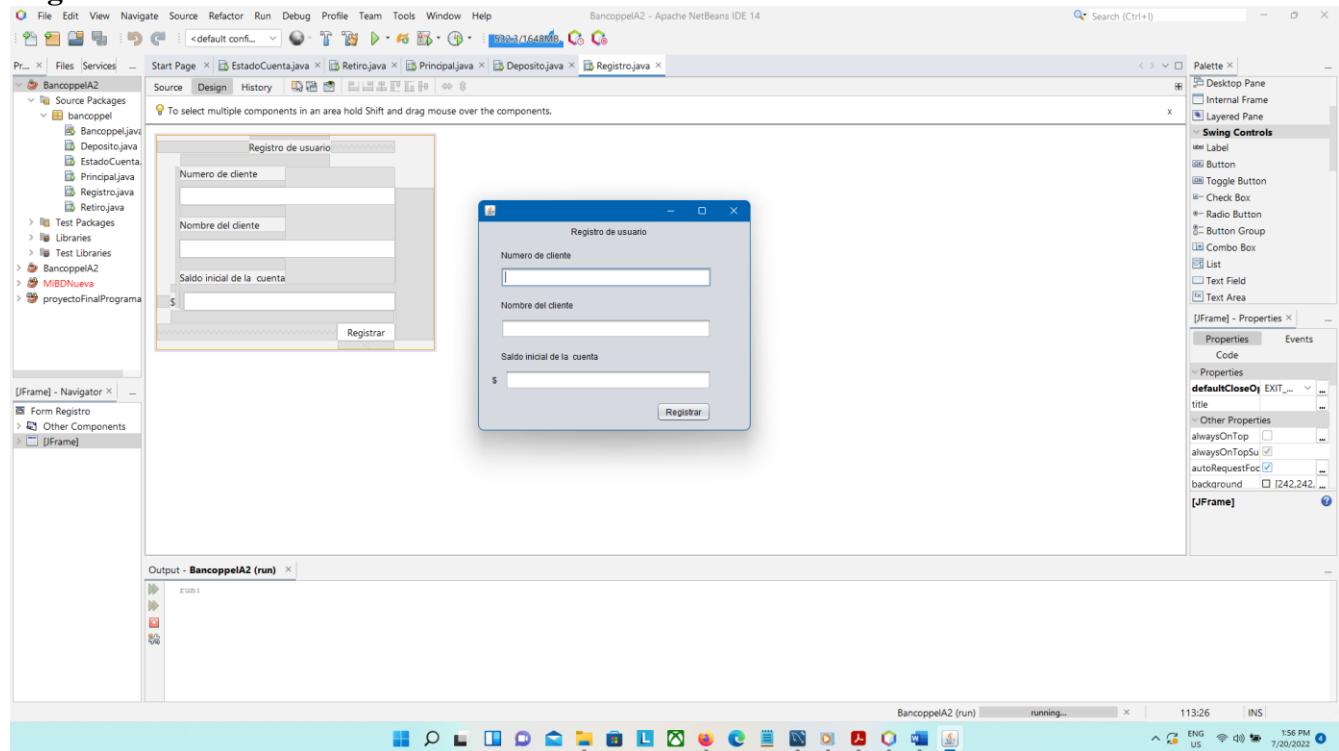
Interfaz.....	pág. 3
Codificación.....	pág. 6
Prueba del Sistema.....	pág. 9
Conclusión.....	pág. 17
Bibliografía.....	pág. 18

## Interfaz

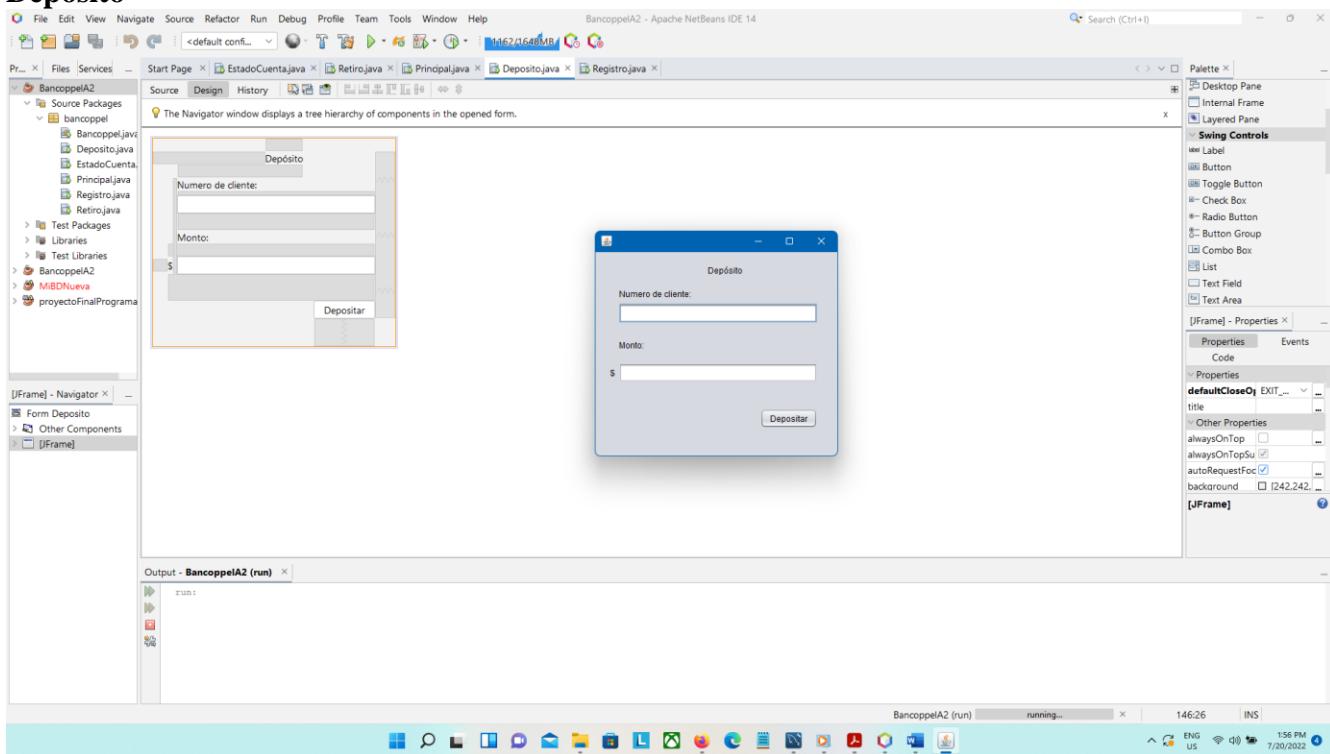
### Pantalla principal



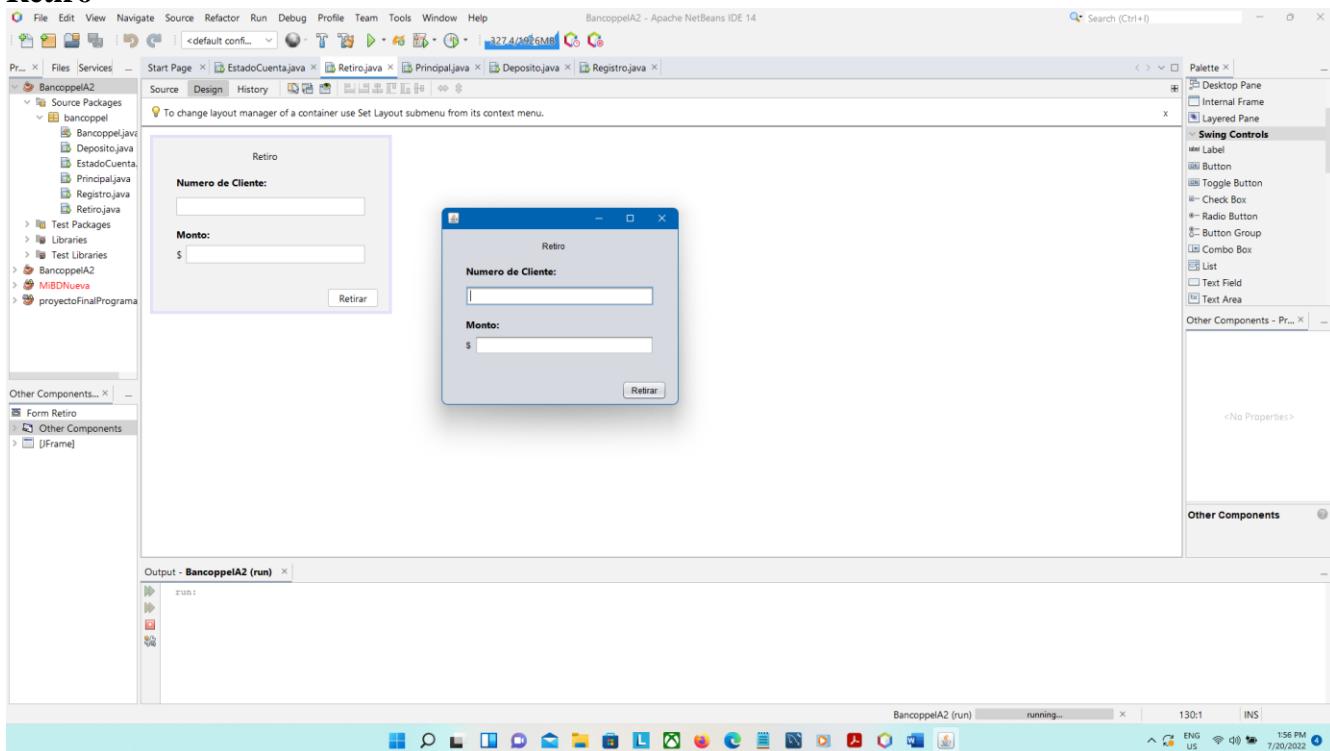
### Registro



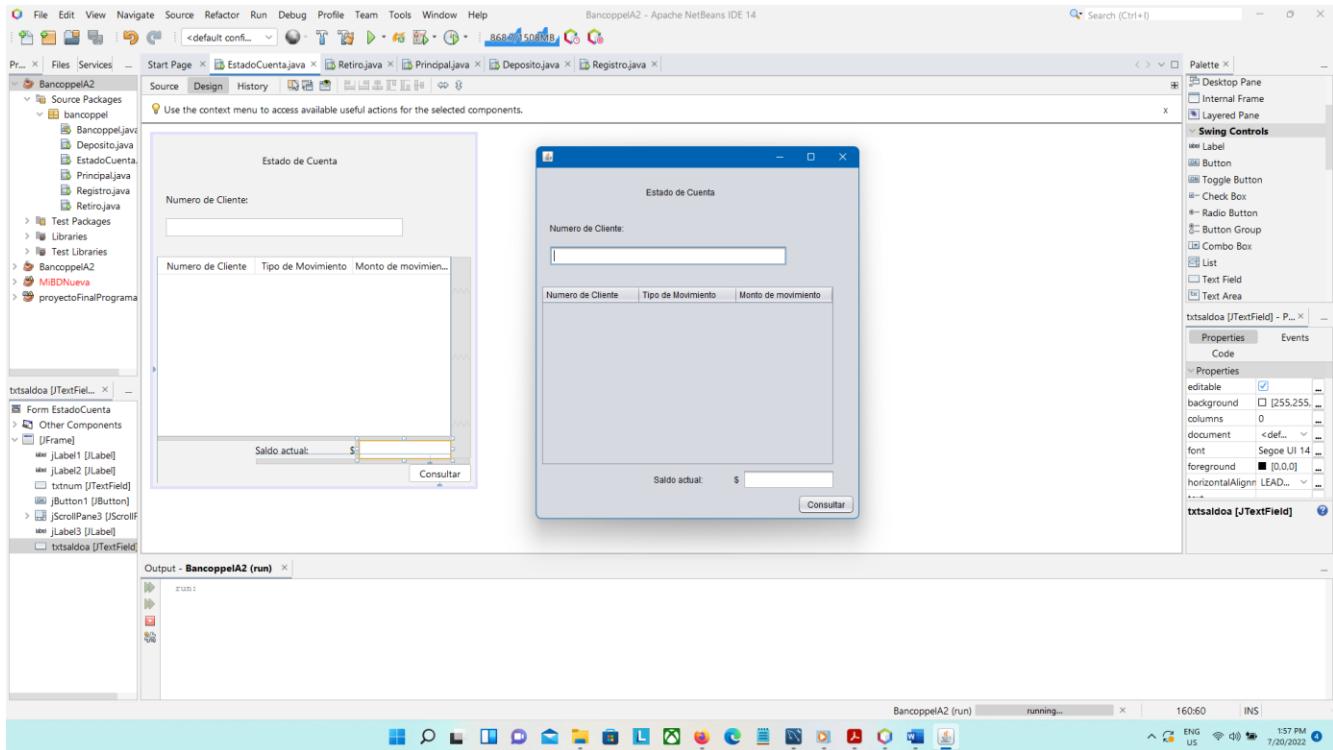
## Deposito



## Retiro



## Consulta de saldo



## Codificación

## Código principal

The screenshot shows the Apache NetBeans IDE interface. The title bar reads "BancoppelA2 - Apache NetBeans IDE 14". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help. The toolbar has icons for file operations like New, Open, Save, Print, Find, and Run. The tabs at the top show "default config...", "EstadoCuenta.java", "Retiro.java", "Principal.java", "Deposito.java", and "Registro.java". The left sidebar shows the Project tree under "BancoppelA2" with packages like "bancoppel" containing classes "Bancoppel.java", "Deposito.java", "EstadoCuenta.java", "Principal.java", "Registro.java", and "Retiro.java". It also lists "Test Packages", "Libraries", "Test Libraries", and "BancoppelA2". The right sidebar shows the History tab. The main area is the code editor for "Principal.java". The code implements a JFrame with components for Registro, Deposito, Retiro, and EstadoCuenta. It includes methods for getting text from JTextField and handling actionPerformed events for JButton1 and JButton2. The Navigator panel on the left lists members of the Principal class, including labels (jLabel1-jLabel5), buttons (jButton1-jButton4), and text fields (txtActionPerformed, txtAtmOp). The status bar at the bottom shows "BancoppelA2 (run)" and "running...".

```
1 package bancoppel;
2
3 import javax.swing.JTextField;
4 public class Principal extends javax.swing.JFrame {
5
6     public Principal() {
7         this.p1 = new Registro();
8         this.p2 = new Deposito();
9         this.p3 = new Retiro();
10        this.p4 = new EstadoCuenta();
11
12        initComponents();
13    }
14
15    Registro p1;
16    Deposito p2;
17    Retiro p3;
18    EstadoCuenta p4;
19
20    @SuppressWarnings("unchecked")
21    // Generated by IntelliJ IDEA
22    private JTextField getTxtAt() {
23        return txtAt;
24    }
25
26    private void txtAtActionPerformed(java.awt.event.ActionEvent evt) {
27        // TODO add your handling code here:
28    }
29
30    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
31        int elec;
32        elec = Integer.parseInt(txtAt.getText());
33
34        switch (elec) {
35            case 1:
36                p1.setVisible(true);
37                break;
38            case 2:
39                p2.setVisible(true);
40                break;
41            case 3:
42                p3.setVisible(true);
43                break;
44            case 4:
45                p4.setVisible(true);
46                break;
47        }
48    }
49
50    // TODO add your handling code here:
51 }
```

# Código de Registro

## Código de Deposito

The screenshot shows the Apache NetBeans IDE interface with the 'Deposito.java' file open in the editor. The code implements a deposit functionality for a bank account. It connects to a MySQL database using JDBC, performs a query to check the current balance, and then updates the balance by the deposited amount. Error handling is included to manage SQL exceptions and non-existent clients.

```
try {
    String driver = "com.mysql.cj.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3308/bancoppel";
    String user = "root";
    String pw = "";
    Class.forName(driver);
    System.out.println("Cargo el driver");
    Connection con = null;
    con = DriverManager.getConnection(url, user, pw);
    System.out.println("conexion correcta");

    Statement st = null;
    st = con.createStatement();
    String num = this.txtnum.getText();
    String mont = this.txtmont.getText();
    double mont2 = Double.parseDouble(this.txtmont.getText());

    ResultSet rs = null;
    String query = "select * from detalle_cuenta_cliente where numero_cliente = " + this.txtnum.getText();
    rs = st.executeQuery(query);
    if (rs.next()) {
        String sql = "INSERT INTO movimientos_cuenta_cliente (numero_cliente, tipo_movimiento ,monto_movimiento) VALUES ('" + num + "','" Deposito' ,'" + mont + "')";
        double saldo = rs.getDouble("saldo_cliente");
        double suma = mont2 + saldo;
        String sql2 = "UPDATE detalle_cuenta_cliente SET saldo_cliente = '" + suma + "' where numero_cliente = " + this.txtnum.getText();
        System.out.println("registro insertado");

        st.executeUpdate(sql);
        st.executeUpdate(sql2);

        System.out.println("deposito realizado, su saldo actual es: " + suma);
    } else {
        JOptionPane.showMessageDialog(null, "cliente no existe");
    }
    con.close();
} catch (ClassNotFoundException | SQLException ex) {
    System.out.println(ex.toString());
}

try {
    String driver = "com.mysql.cj.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3308/bancoppel";
    String user = "root";
    String pw = "";
    Class.forName(driver);
    System.out.println("Cargo el driver");
    Connection con = null;
```

## Código de Retiro

The screenshot shows the Apache NetBeans IDE interface with the 'Retiro.java' file open in the editor. The code implements a withdrawal functionality for a bank account. It connects to a MySQL database using JDBC, performs a query to check the current balance, and then updates the balance by the withdrawn amount. Error handling is included to manage SQL exceptions and non-existent clients.

```
try {
    String driver = "com.mysql.cj.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3308/bancoppel";
    String user = "root";
    String pw = "";
    Class.forName(driver);
    System.out.println("Cargo el driver");
    Connection con = null;
    con = DriverManager.getConnection(url, user, pw);
    System.out.println("conexion correcta");

    Statement st = null;
    st = con.createStatement();
    String num = this.txtnum.getText();
    String mont = this.txtmont.getText();
    double mont2 = Double.parseDouble(this.txtmont.getText());

    ResultSet rs = null;
    String query = "select * from detalle_cuenta_cliente where numero_cliente = " + this.txtnum.getText();
    rs = st.executeQuery(query);

    if (rs.next()) {
        double saldo = rs.getDouble("saldo_cliente");
        double resta = saldo - mont2;

        if (mont2 <= saldo) {
            String sql = "INSERT INTO movimientos_cuenta_cliente (numero_cliente, tipo_movimiento ,monto_movimiento) VALUES ('" + num + "','" Retiro' ,'" + mont + "')";
            st.executeUpdate(sql);
            //-----
            String sql2 = "UPDATE detalle_cuenta_cliente SET saldo_cliente = '" + resta + "' where numero_cliente = " + this.txtnum.getText();
            System.out.println("registro insertado");

            st.executeUpdate(sql2);

            System.out.println("Retiro realizado, su saldo actual es: " + resta);
        } else {
            JOptionPane.showMessageDialog(null, "Error: fondos insuficientes");
        }
        con.close();
    } catch (ClassNotFoundException | SQLException ex) {
        System.out.println(ex.toString());
    }
} catch (SQLException ex) {
    System.out.println("Error: " + ex.getMessage());
}
```

## Código de Estado de Cuenta

```
try {
    String driver = "com.mysql.cj.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3306/bancoppel";
    String user = "root";
    String pw = "";

    Class.forName(driver);
    System.out.println("Cargo el driver");
    Connection con = null;

    con = DriverManager.getConnection(url, user, pw);
    System.out.println("conexion correcta");

    Statement st = null;
    st = con.createStatement();
    ResultSet rs = null;
    Object[] Cliente = new Object[3];
    Object[] detalle_cuenta_cliente = new Object[3];
    Cliente = (DefaultTableModel) TablaA.getModel();
    detalle_cuenta_cliente = (DefaultTableModel) TablaB.getModel();

    String query = "select numero_cliente from detalle_cuenta_cliente where numero_cliente= " + this.txtnum.getText();

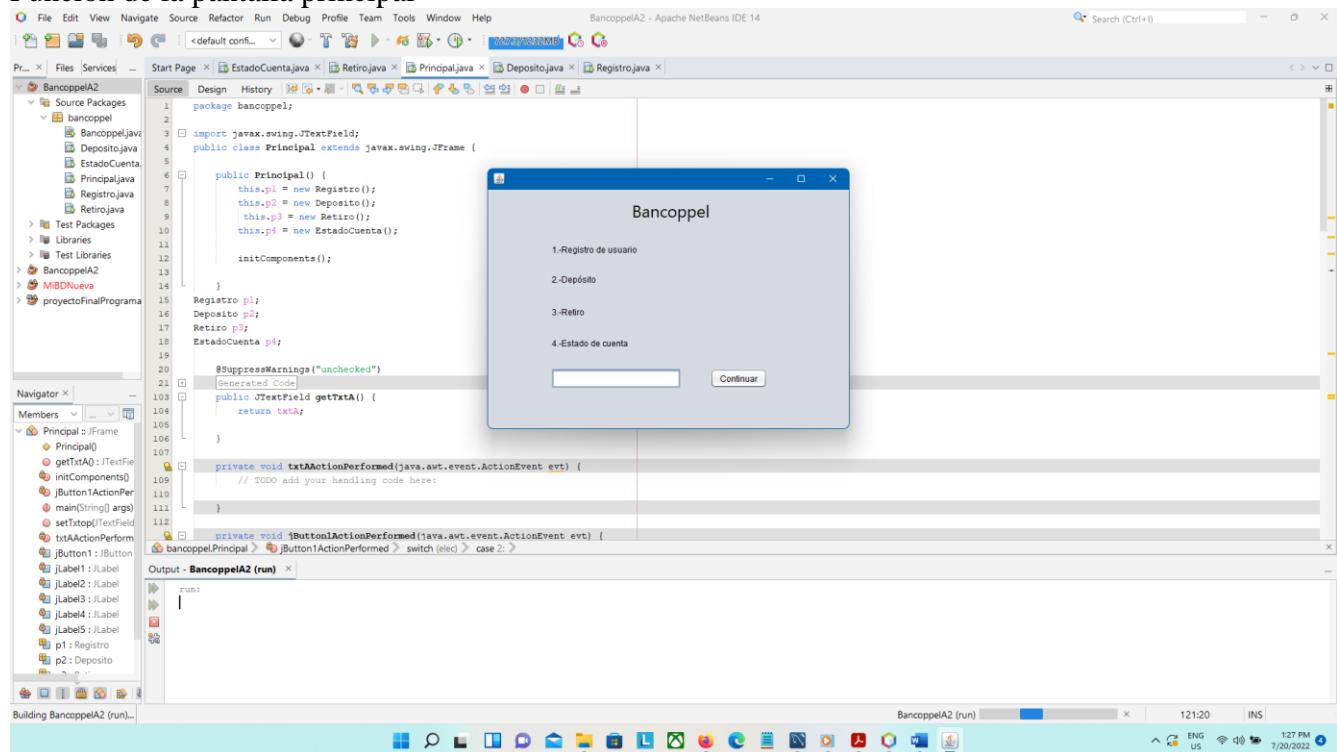
    rs = st.executeQuery(query);
    if (rs.next()) {

        while (rs.next()) {
            Cliente[0] = rs.getInt("numero_cliente");
            Cliente[1] = rs.getString("tipo_movimiento");
            Cliente[2] = rs.getDouble("monto_movimiento");
            modelo.addRow(Cliente);
        }
        TablaA.setModel(modelo);
    } else {
        JOptionPane.showMessageDialog(null, "Error: no hay movimientos que mostrar");
    }

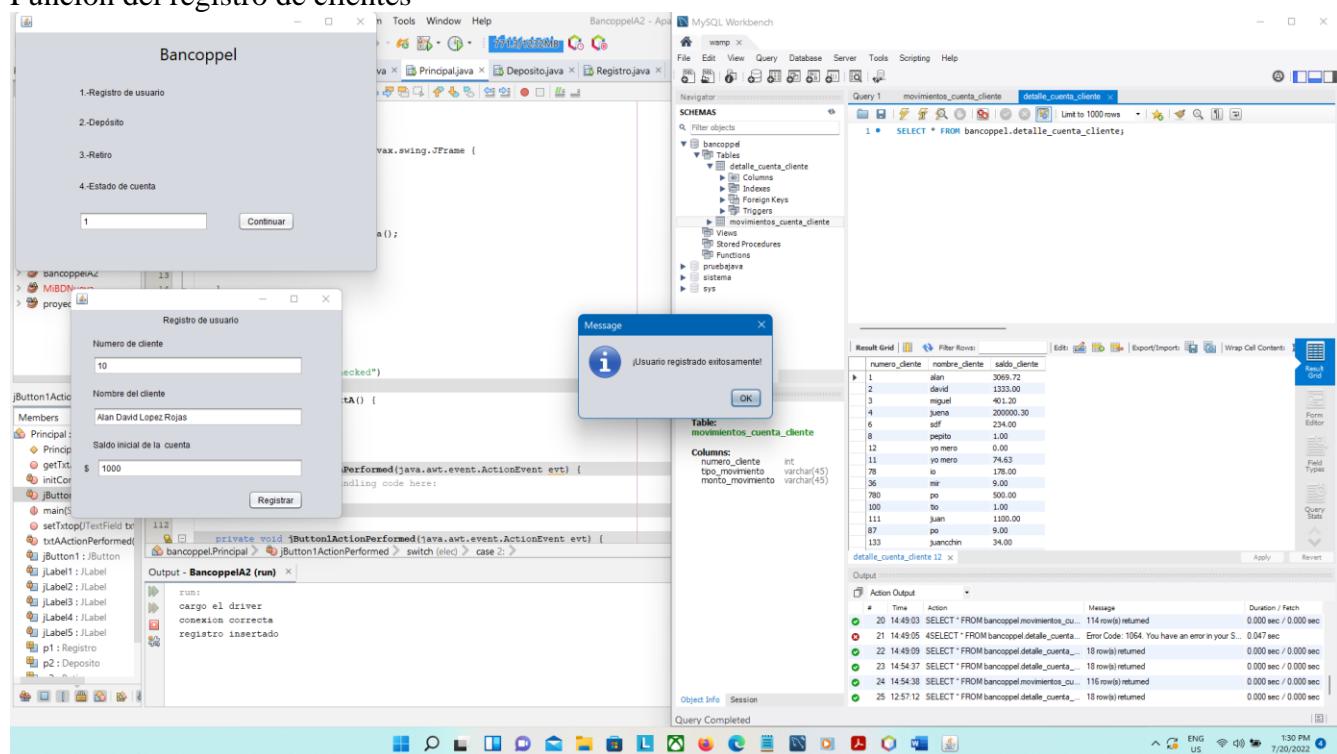
} catch (SQLException ex) {
    System.out.println(ex.toString());
}
}
```

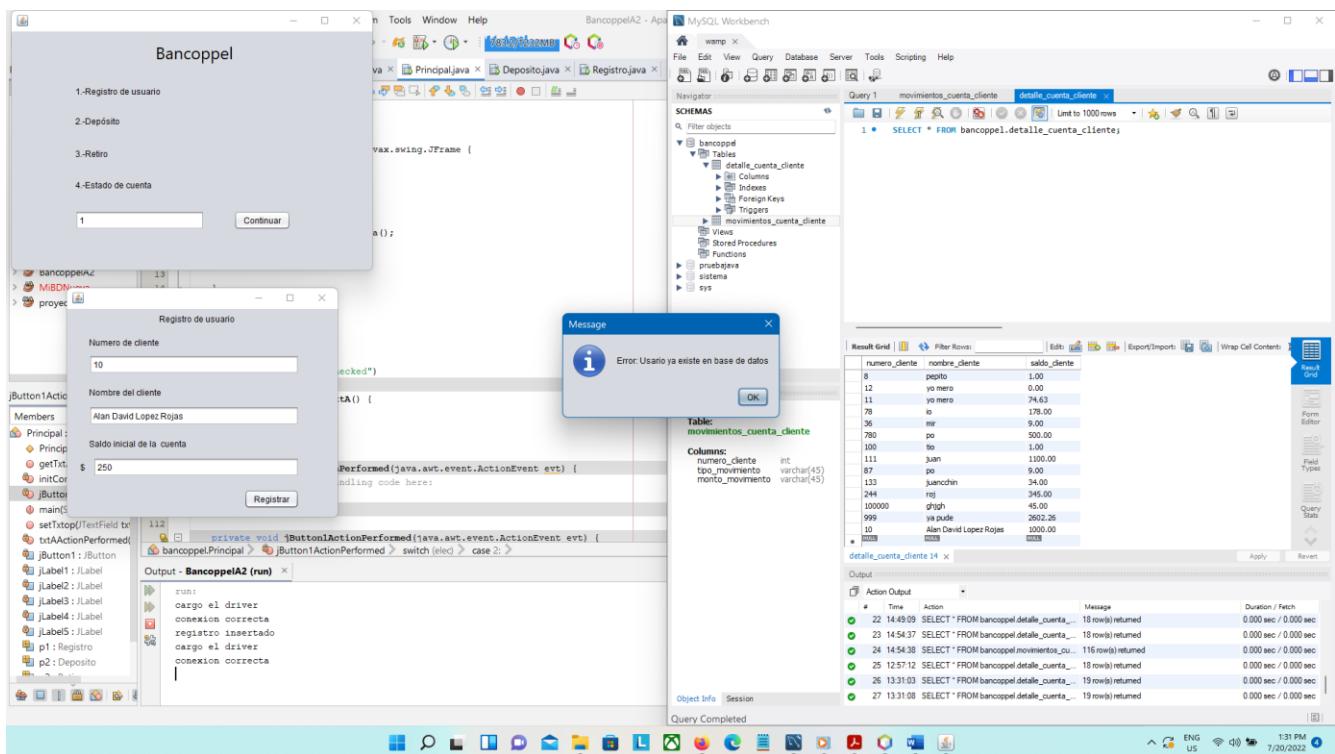
## Prueba del sistema

### Función de la pantalla principal

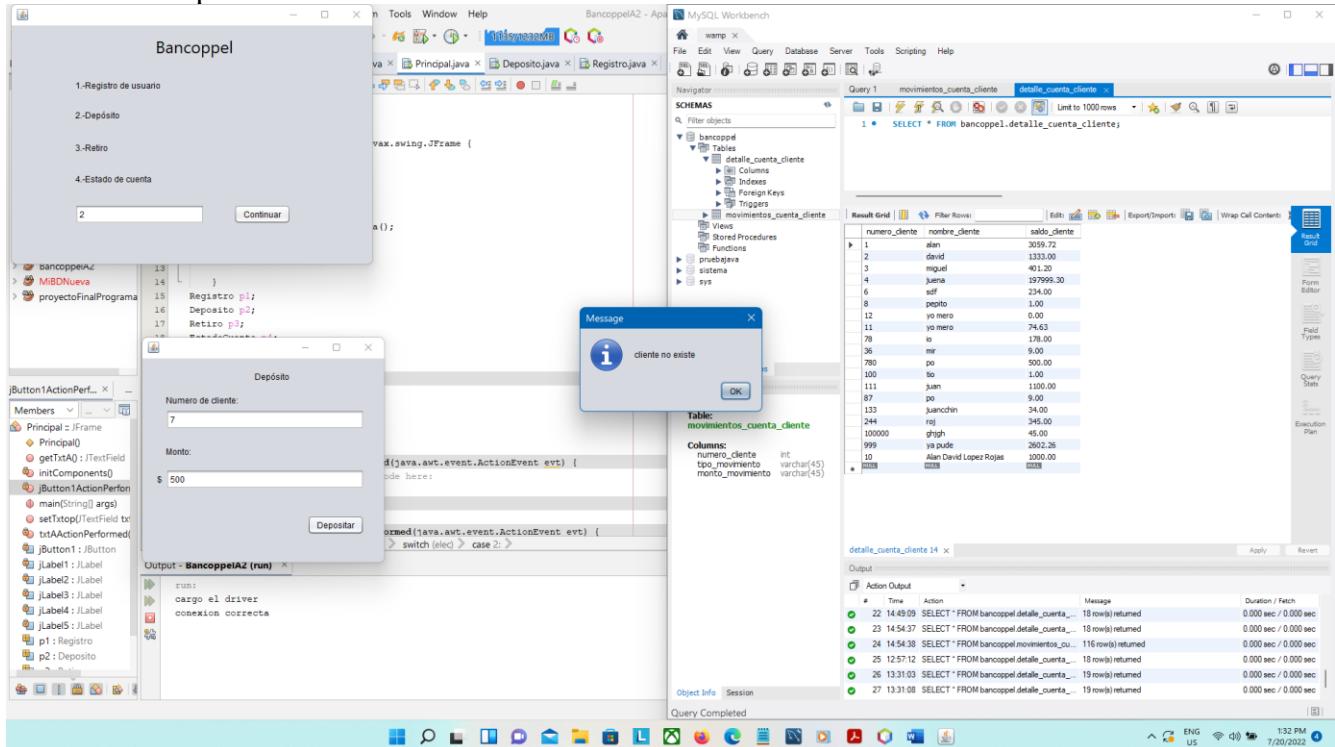


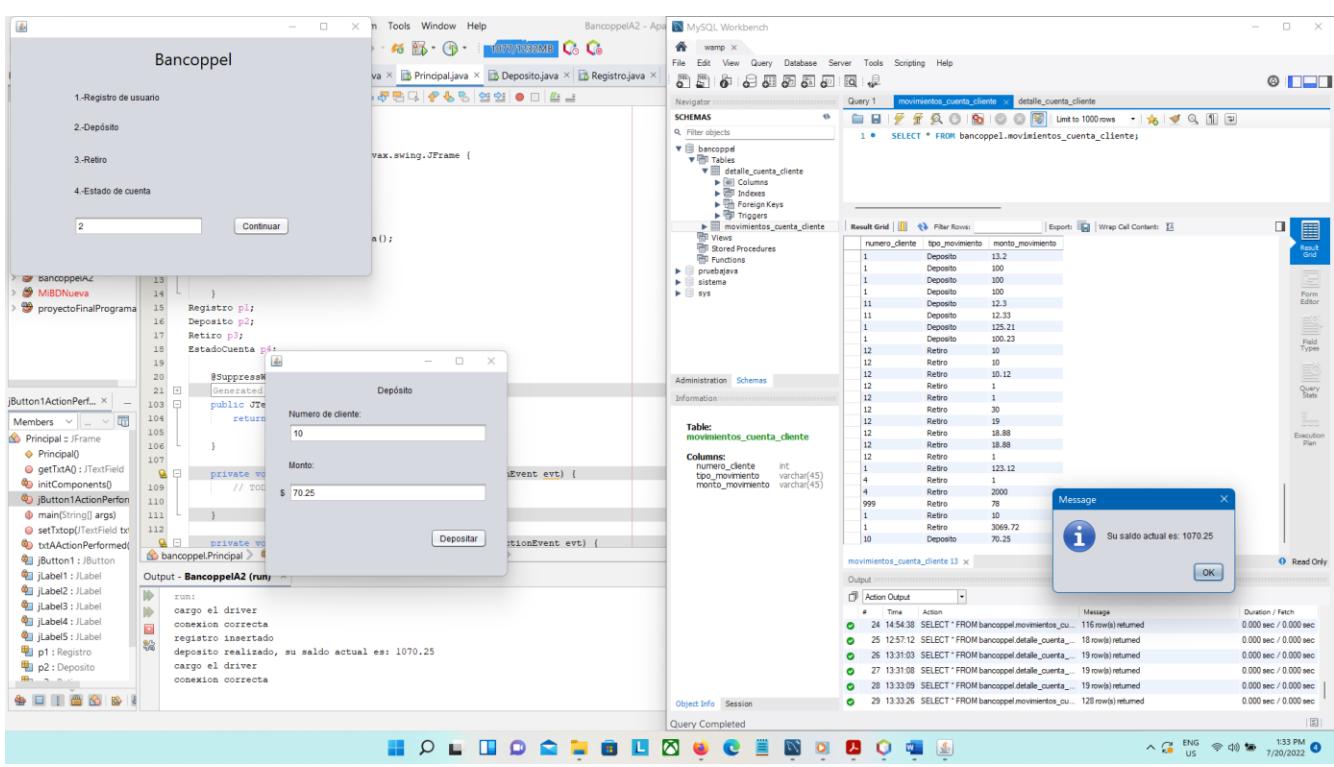
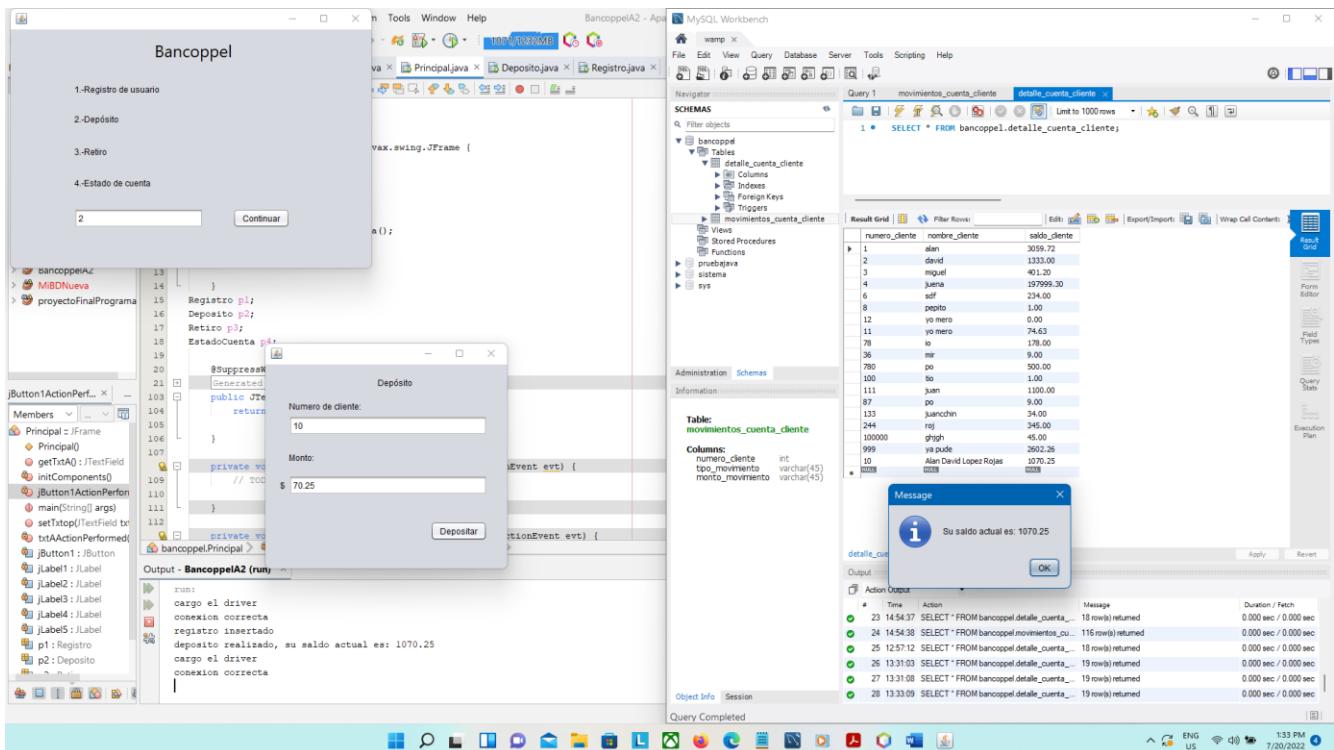
### Función del registro de clientes



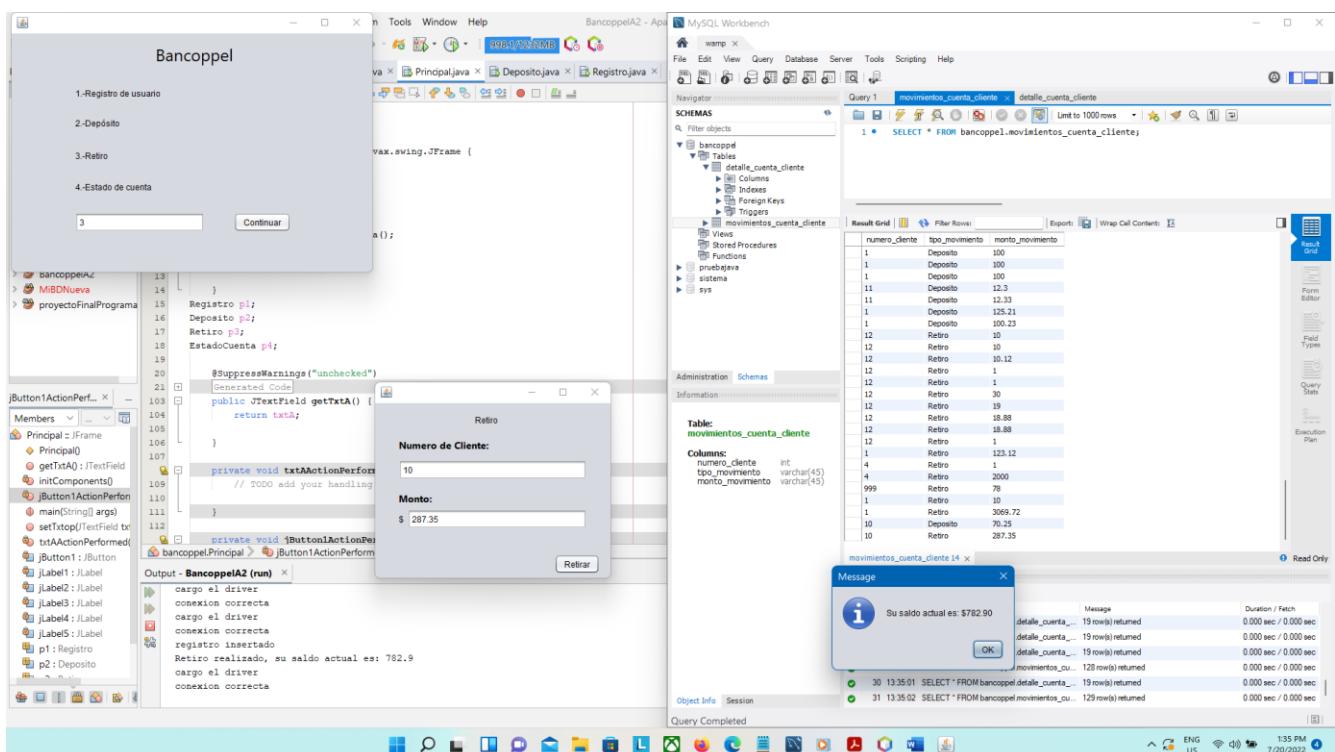
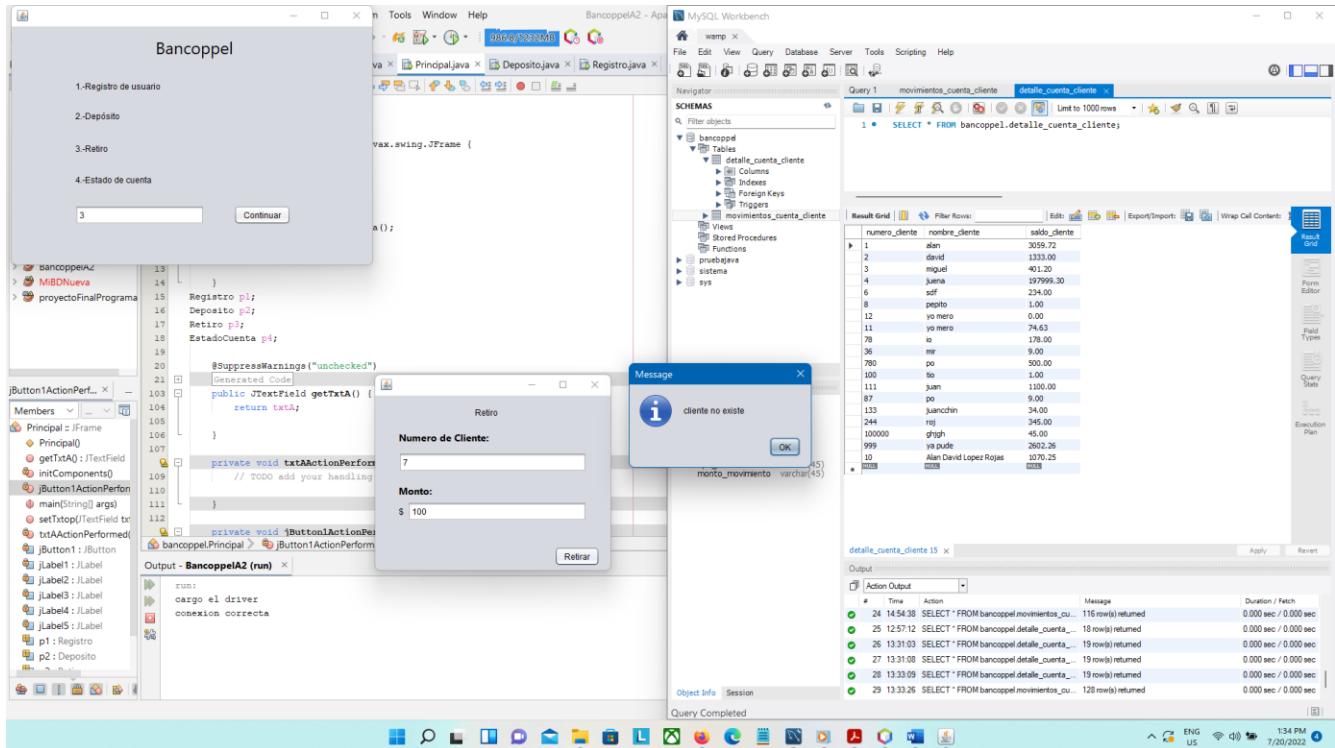


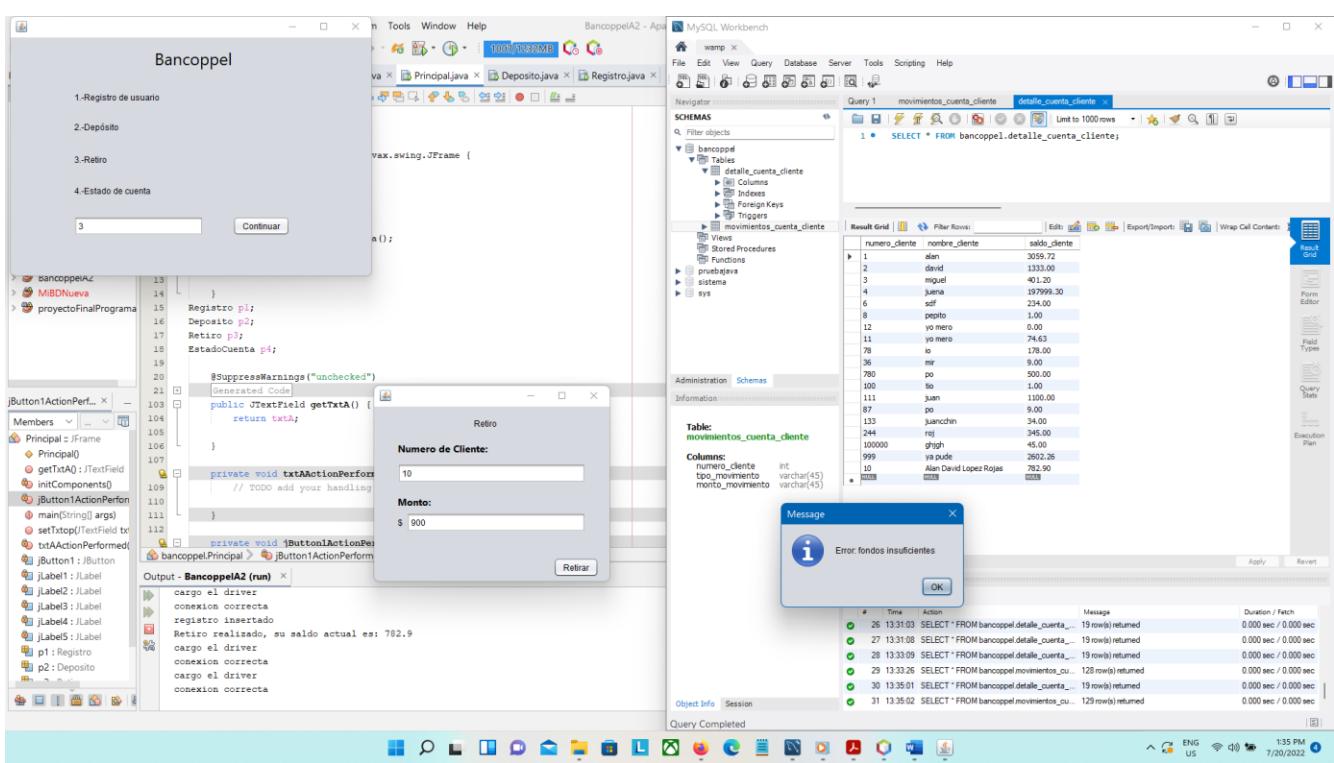
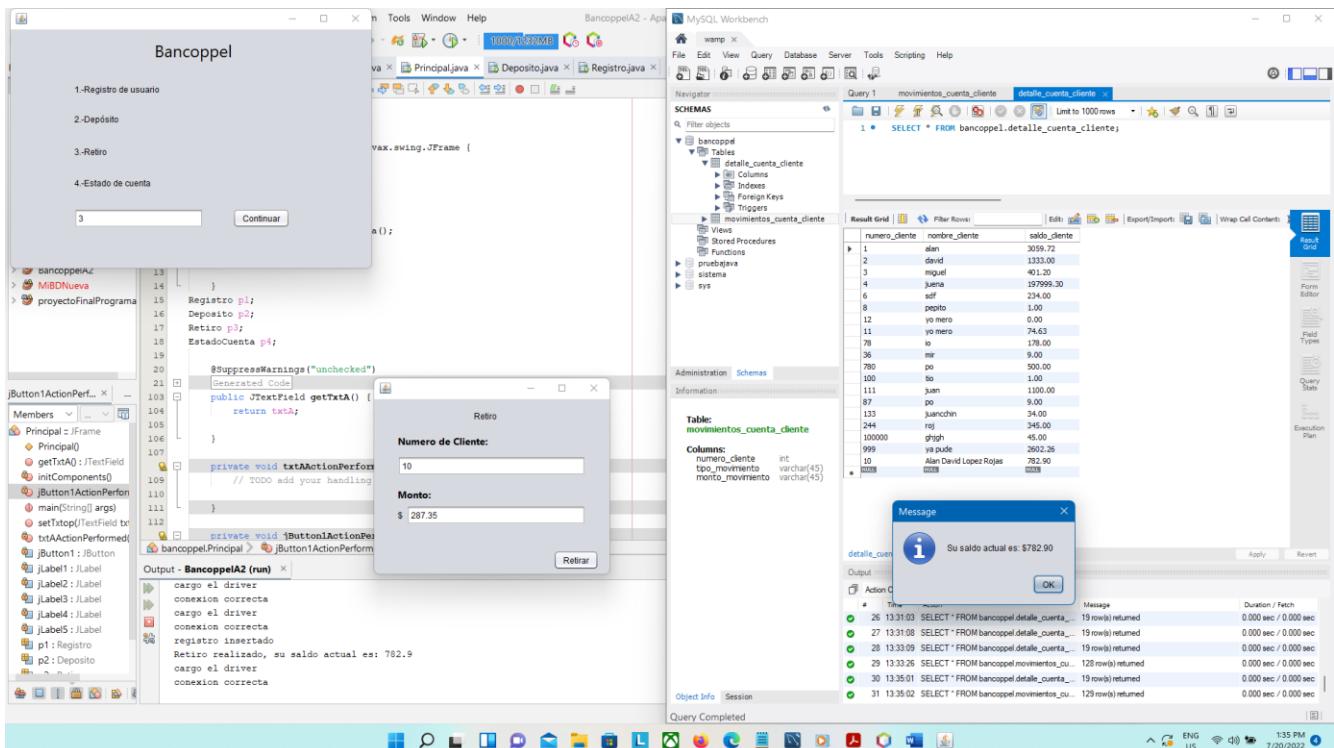
## Función del depósito



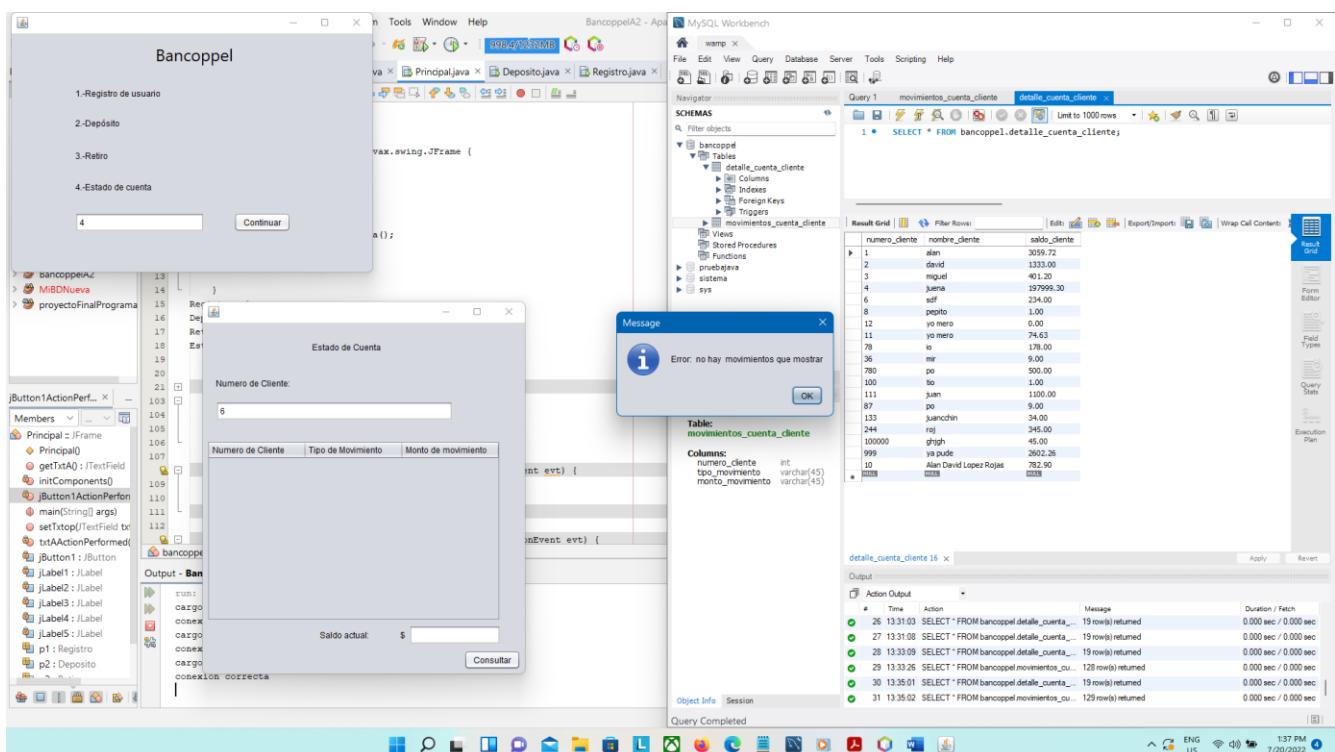
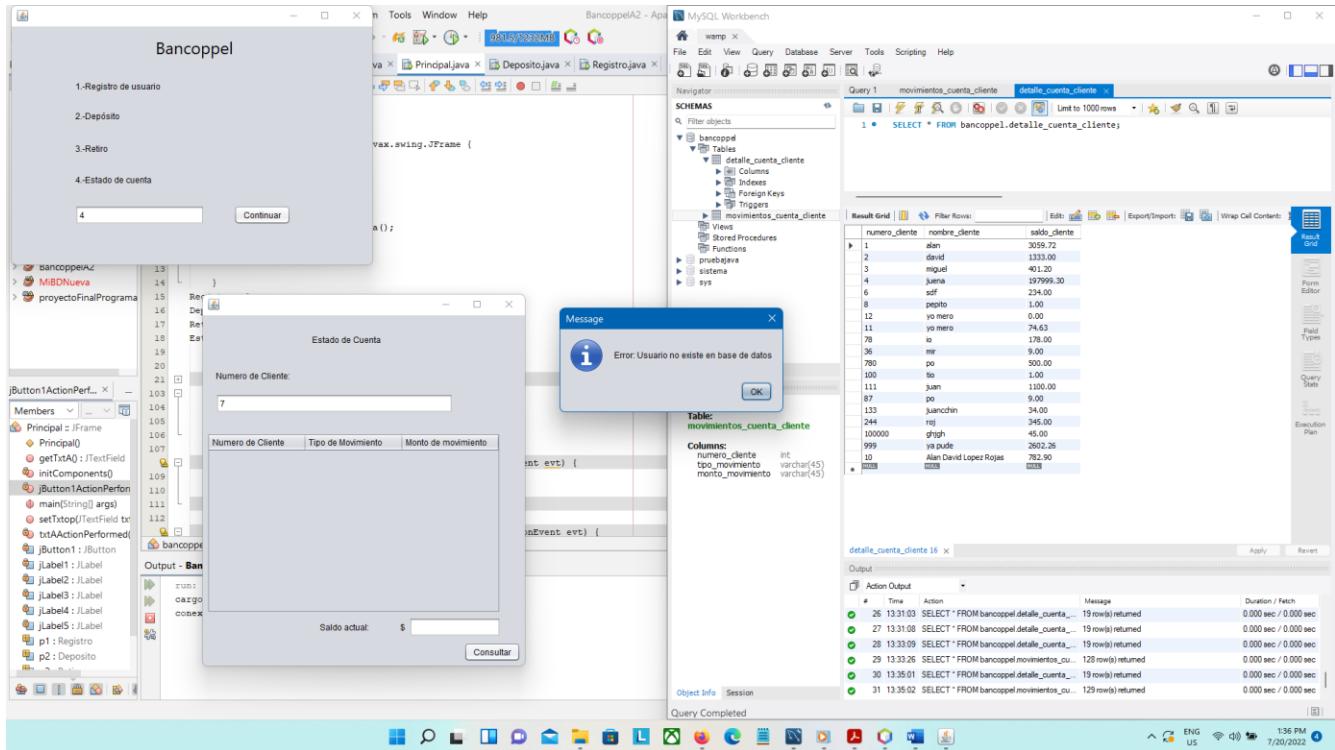


## Función del retiro





## Función de la consulta de clientes



Bancoppel

1-Registro de usuario  
2-Depósito  
3-Retiro  
4-Estado de cuenta

4 Continuar

Numero de Cliente: 10

Numero de Cliente	Tipo de Movimiento	Monto de movimiento
10	Retiro	287.35
10	Retiro	900.0
10	Retiro	4000.0
10	Retiro	10000.0
10	Depósito	10000.0

Saldo actual: \$ 782.90 Consultar

MySQL Workbench

Query 1: SELECT \* FROM bancoppel.detalle\_cuenta\_cliente;

numero_cliente	nombre_cliente	saldo_cliente
1	alan	3059.72
2	david	1333.00
3	miguel	401.20
4	juana	197999.30
6	edf	234.00
8	pepe	1.00
12	yo mero	0.00
11	yo mero	74.63
78	io	178.00
36	mir	9.00
79	oo	990.00
100	lo	1.00
111	juan	1100.00
87	po	9.00
133	juanchin	34.00
244	roj	345.00
100000	grph	45.00
999	ia pude	3602.26
10	Alan David Lopez Rojas	782.90

Output - BancoppelA2 (run) x 130:1 INS

Object Info Session Query Completed

13:43:52 7/20/2022 1:45 PM ENG US

Bancoppel

1-Registro de usuario  
2-Depósito  
3-Retiro  
4-Estado de cuenta

4 Continuar

Numero de Cliente: 10

Numero de Cliente	Tipo de Movimiento	Monto de movimiento
10	Retiro	287.35
10	Retiro	900.0
10	Retiro	4000.0
10	Retiro	10000.0
10	Depósito	10000.0

Saldo actual: \$ 782.90 Consultar

MySQL Workbench

Query 1: SELECT \* FROM bancoppel.movimientos\_cuenta\_cliente;

numero_cliente	tipo_movimiento	monto_movimiento
1	Deposito	125.21
1	Deposito	100.23
12	Retiro	10
12	Retiro	10
12	Retiro	10.12
12	Retiro	1
12	Retiro	1
12	Retiro	30
12	Retiro	10
12	Retiro	18.88
12	Retiro	18.88
12	Retiro	1
1	Retiro	123.12
4	Retiro	1
4	Retiro	2000
999	Retiro	9
1	Retiro	10
1	Retiro	3069.72
10	Deposito	70.25
10	Retiro	287.35
10	Retiro	900
10	Retiro	4000
10	Retiro	10000
8	Retiro	100
10	Deposito	10000

Output - BancoppelA2 (run) x 130:1 INS

Object Info Session Query Completed

13:41:37 7/20/2022 1:45 PM ENG US



## **Conclusión**

Como conclusión final acerca del desarrollo de este proyecto final me dejó muchos aprendizajes, ya que tuve que hacer búsquedas de distintas fuentes que me dieran una idea de cómo desarrollar un sistema de tipo banco y aplicarlo a lo requerido de la actividad, por lo que pude ver distintas formas de solucionar un problema de código o como se realiza la conexión entre JAVA y MySQL, con esto pude comprender mejor el trabajo de un desarrollador de software y como este tipo de trabajo tiene múltiples aplicaciones en muchas áreas , desde este ejemplo de programa para Coppel de un banco hasta crear un sistema de registro y manejo de cuentas para una tienda pequeña, esto solo es una pequeña parte todo lo que implica manejar java.

Nota: Link de Drive descargar para código del proyecto

[https://drive.google.com/file/d/1urOxeChm2-WaO0bCZENBHZmaaCIppIW\\_/view?usp=sharing](https://drive.google.com/file/d/1urOxeChm2-WaO0bCZENBHZmaaCIppIW_/view?usp=sharing)

Link de Drive para descargar video de muestra de función del proyecto final

<https://drive.google.com/file/d/1IAAnqVRZqfhi7tXKNwUswhwnRdRyUJ3ol/view?usp=sharing>

## **Bibliografía**

(Barnes, 2007)

Barnes, D. J. (2007).

Programación Orientada a Objetos con Java. Una Introducción Práctica Usando BlueJ . [[VitalSource Bookshelf version]]. Retrieved from vbk://9788483223505