

Actividad |3| Programa Banco Mexicano Parte 2.

Lenguajes de Programación IV.

Ingeniería en Desarrollo de Software.



TUTOR: Aarón Iván Salazar Macías.

ALUMNO: Ramón Ernesto Valdez Felix.

FECHA: 09/08/2024.

Introducción.	3
Descripción.	3
Justificación.	4
Desarrollo:	5
Interfaz.	5
Codificación.	9
Conclusion.	16
Referencias.	16

Introducción.

En esta tercera actividad de la materia de Lenguajes de la Programación IV, nos planteamos realizar la documentación de la segunda fase de la conexión de la base de datos de la aplicación del banco mexicano que fue creada en la primera fase, esa servirá para que el clientes y empleados del banco de México puedan realizar depósitos de efectivo, retiros y consultas de saldo. Por recomendación del maestro de la materia se decidió tomar el curso proporcionado donde dice que nos basemos en su ejemplo para la realización de esta actividad, donde la recomendación es realizar la interfaz de la aplicación con 4 ventanas que son las siguientes: 1.-Menú principal, 2.-Depósito, 3.-Retiro, 4.-Saldo o consulta y por último hacer la conexión a la base de datos. A continuación anexamos la información del material de la actividad 2 que nos sirve para la actividad final de la materia como contexto adicional: los clientes de Banco Mexicano necesitan un programa que les permita a sus clientes el realizar depósitos, retiros y consultas de su saldo. Por lo que necesitan que un ingeniero en desarrollo de software genere una base de datos que atienda a esta necesidad.

Descripción.

En esta actividad numero tres de la materia de Lenguajes de la Programación IV, realizaremos la documentación, la conexión de la base de datos a la aplicación de transacciones bancarias solicitada con las transacciones comunes que realiza un clientes y empleados en los bancos mexicanos, se nos pide realizar la interfaz y conectarla a su base de datos, ya que seremos el desarrollador de software contratado por el proveedor de servicio del banco el cual nos pide realizar lo siguiente: se necesita un programa que les permita a sus clientes o empleados el realizar depósitos, retiros y consultas de su saldo. Por lo que el desarrollo de software genera una base de datos que atienda a esta necesidad. Esta información será llevada al lenguaje de programación de Java con la herramienta Apache NetBeans que se propone en la documentación de la actividad así programando la aplicación de banco mexicano que solicitan en la materia en curso así teniendo el derecho a una calificación para continuar en la carrera de IDS.

Justificación.

En esta actividad trabajaremos con la documentación de la conexión de la estructura de la interfaz creación en la primera parte de la actividad, aplicación del banco mexicano donde permite realizar las transacciones básicas del banco con ayuda del cajero personalizado o automático, las transacciones básicas que nos debe permitir realizar la aplicación son retiró, depósito y consulta de saldo, utilizar el lenguaje Java 8 donde la pantalla principal debe contar con un menú de las las opciones mencionadas con anterioridad y el entorno de programación sugerido realizar el programa con los siguientes requerimientos:

- Menú de Interfaz principal:
 - Depósito.
 - Retiro.
 - Saldo.
 - Salir
- Pantalla Deposito.
 - Monto a Depocitar.
 - Boton de Continuar.
- Pantalla Retiro.
 - Monto de Retio.
 - Boton de Continuar.
- Pantalla Saldo.
 - Consulta de saldo.
 - Boton de Continuar.
- Utilizar la herramienta Apache NetBeans con complemento de Java 8.
- Utilizar sentencias IF, FOR, Switch-case entre otras, según sea conveniente.
- Agregar a GitHub la calculadora IMC realizada.

5

- PDF de esta actividad en el portafolio GitHub.
- Anexa link de GitHub en documento.

Desarrollo:

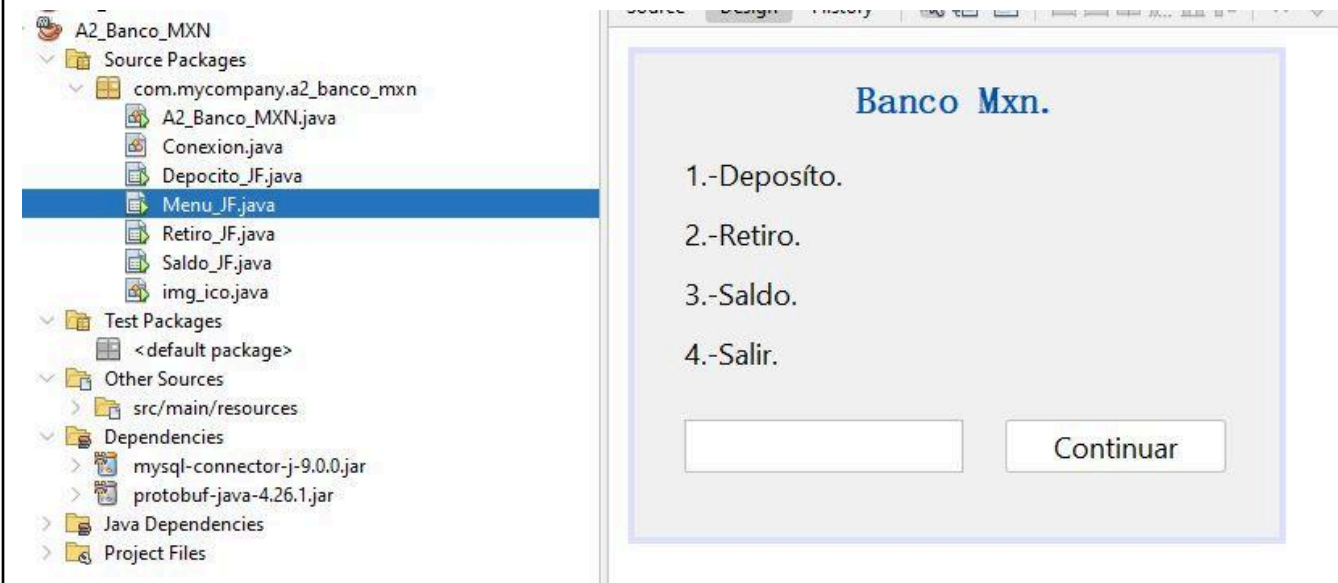
En este punto realizaremos la conexión de la base de datos con el desarrollo de la aplicación del banco de México para que al cliente le permitirá utilizar el cajero personalizado o bien automático donde permitirá realizar cualquiera de las transacciones disponibles en el desarrollo de la aplicación del banco, a continuación realizaremos la documentación de la interfaz y el desarrollo del código.

Link: [GitHub](#)

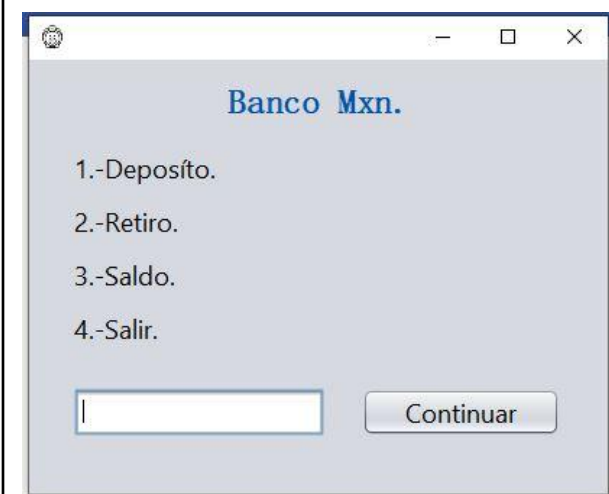
Interfaz.

En este punto de la actividad realizaremos captura de pantalla de la interfaz de cómo quedó al finalizar la última parte de la aplicación del banco mexicano, sufrió unos pequeños cambios que no se deben de notar mucho como el retirar el símbolo \$ en los cuadros de texto y la pantalla retiro y saldo sufrieron un cambio de color a las pantallas entregadas en la primera parte de la actividad, adicional cabe mencionar que se dará una breve explicación de la pantalla de aplicación a mostrar en dicha documentación, por tal motivo no creo que sea apto para la distribución en algún hospital de la república mexicana.

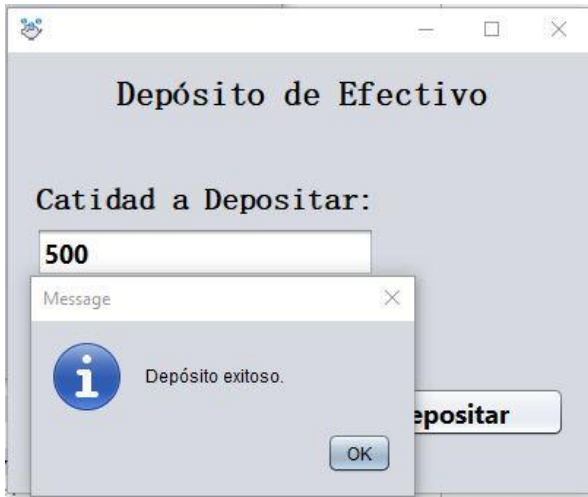
Captura de pantalla del Menú Principal



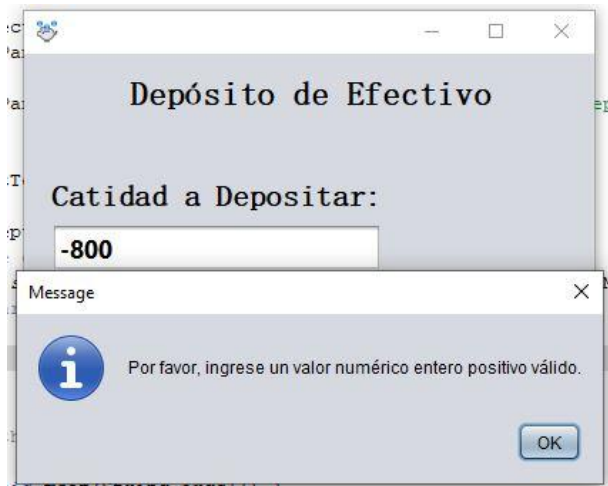
En esta pantalla mostramos el proyecto de la actividad final de la aplicación del banco mexicano, donde nos muestra la pantalla principal seleccionada y en su menú expandible de la aplicación muestra toda la estructura de la aplicación.



En esta pantalla mostramos la ejecución de la pantalla principal de la aplicación del banco mexicanos, mostrando los datos mencionado en la pantalla anterior.

Captura de pantalla de deposito.

Esta es la pantalla de la interfaz en ejecución de depósito, mostramos como funciona la aplicación al hacer de forma exitosa el depósito de \$500 pesos.



En esta segunda pantalla de la interfaz en ejecución de depósito, mostramos el mensaje de error al tratar de realizar un depósito de manera incorrecta utilizando números negativos .

Captura de pantalla de Retiro.

Esta es la pantalla de la interfaz en ejecución de retiro, mostramos como funciona de manera correcta la aplicación al hacer de forma exitosa el retiro de \$1100 pesos.

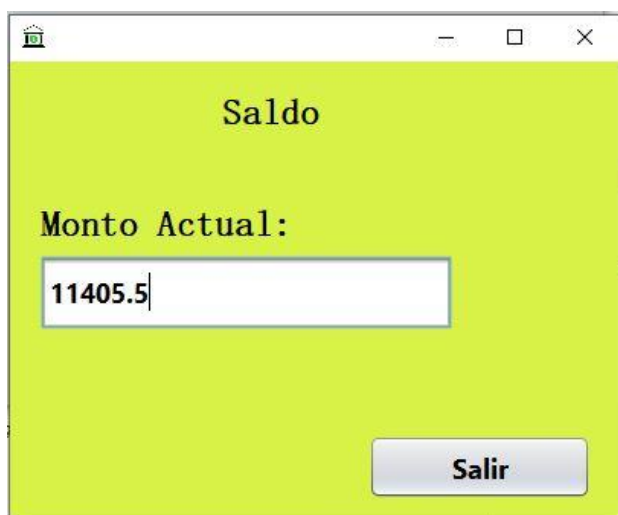


En esta segunda pantalla de la interfaz en ejecución de retiro, mostramos el mensaje de error al tratar de realizar un retiro de manera incorrecta utilizando números negativos.

Captura de pantalla de Saldo.



Esta es la pantalla de la interfaz de saldo de la aplicación del banco mexicanos, donde nos muestra el cuadro de texto de monto actual de la cuenta consultada.



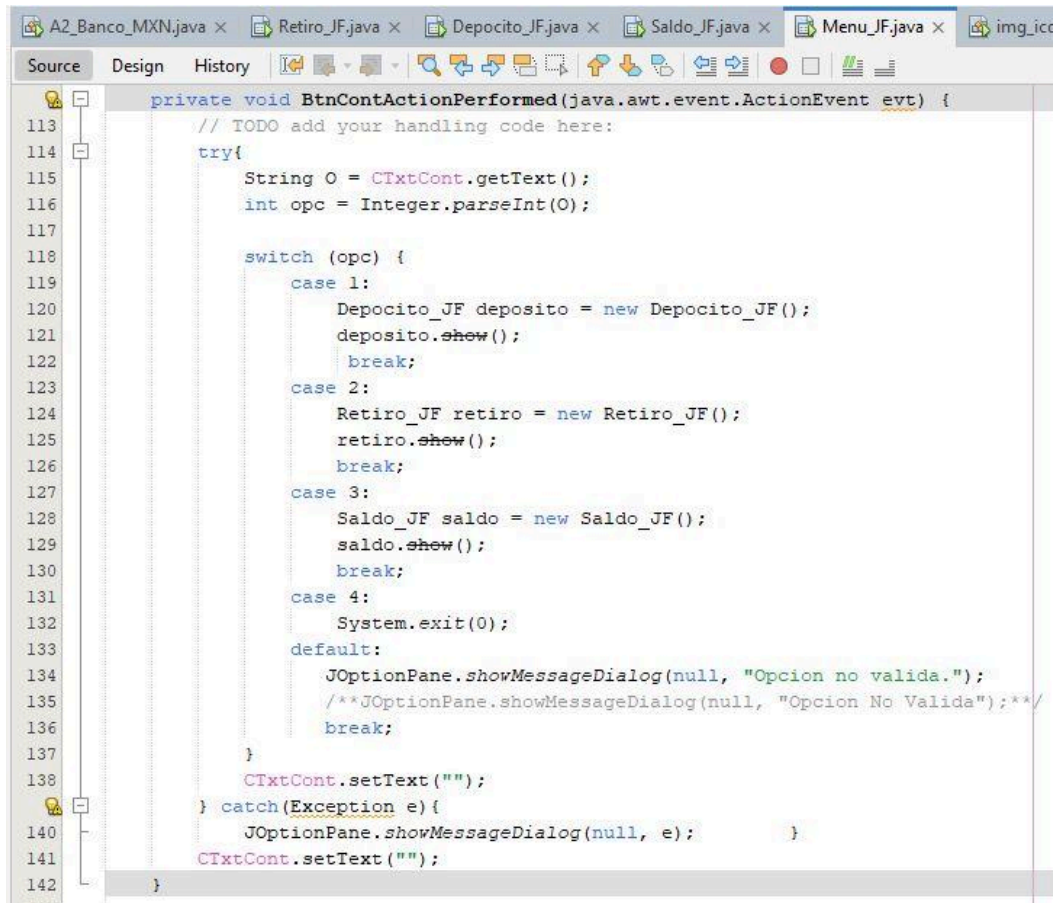
En esta pantalla mostramos la ejecución de la pantalla Saldo con un monto diferente al anterior ya que realizaron prueba de depósitos y retiros: con ingreso de números negativos y decimales por dicho motivo el diferente monto mostrado.

En estas app realizamos la actividad recomendada por el docente de la materia de Lenguajes de la Programación IV.

Codificación.

En este punto daremos un breve explicacion del codigo que se anexara en las imagenes de evidencia, esto para dar más claridad de lo que se realizó para llegar a la creación de la aplicación del banco mexicano solicitada en la actividad a continuación daremos el detalle:

Captura de pantalla del código del Menú Principal.



```

private void BtnContActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        String O = CTxtCont.getText();
        int opc = Integer.parseInt(O);

        switch (opc) {
            case 1:
                Depocito_JF deposito = new Depocito_JF();
                deposito.show();
                break;
            case 2:
                Retiro_JF retiro = new Retiro_JF();
                retiro.show();
                break;
            case 3:
                Saldo_JF saldo = new Saldo_JF();
                saldo.show();
                break;
            case 4:
                System.exit(0);
            default:
                JOptionPane.showMessageDialog(null, "Opcion no valida.");
                /**JOptionPane.showMessageDialog(null, "Opcion No Valida");**/
                break;
        }
        CTxtCont.setText("");
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e);
    }
    CTxtCont.setText("");
}

```

Pantalla código botón de la interfaz Menú Principal: Este fragmento de código Java, que se muestra a continuación, se ejecuta cuando el usuario hace clic en un botón etiquetado como 'Continual' en la interfaz gráfica. Su función principal es abrir la pantalla del menú seleccionado en el cuadro de texto que existe en la interfase, como depósito, retiro y saldo.



```

import javax.swing.JOptionPane;
import javax.swing.ImageIcon;

/**
 *
 * @author ramon.valdez
 */
public class Menu_JF extends javax.swing.JFrame {

    /**
     * Creates new form Menu_JF
     */
    public Menu_JF() {
        initComponents();
        setIconImage( new ImageIcon(getClass().getResource("/pig.PNG")).getImage());
    }
}

```

Captura de pantalla del código del Depósito.

```
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;

/**
 *
 * @author ramon.valdez
 */
public class Depocito_JF extends javax.swing.JFrame {

    /**
     * Creates new form Depocito_JF
     */
    public Depocito_JF() {
        initComponents();
        setIconImage( new ImageIcon(getClass().getResource("/Oink.PNG")).getImage());
    }
}
```

Código de pantalla de Depósito: Solo mostramos un pequeño código que muestra las librerías utilizadas y la línea donde agregamos un icono en la parte de arriba de la aplicación al ejecutar la app.

```

private void BtnDepActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        Connection cnc = new Conexion().getConnection();
        PreparedStatement ps = cnc.prepareStatement("UPDATE cuenta SET saldo = saldo + ? WHERE id = 1");

        // Validar el valor ingresado
        String depositoStr = CtxtDepo.getText();
        int deposito;

        // Verificar si el valor es un número entero positivo
        try {
            deposito = Integer.parseInt(depositoStr);
            if (deposito <= 0) {
                throw new NumberFormatException("El valor debe ser un número entero positivo.");
            }
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(null, "Por favor, ingrese un valor numérico entero positivo válido.");
            return; // Salir del método si no es un número entero positivo
        }

        ps.setInt(1, deposito);

        int rowsAffected = ps.executeUpdate();

        if (rowsAffected > 0) {
            JOptionPane.showMessageDialog(null, "Depósito exitoso.");
        } else {
            JOptionPane.showMessageDialog(null, "Error al realizar el depósito.");
        }

        CtxtDepo.setText("");
        cnc.close();
    } catch (SQLException e) {
        // Manejo de excepciones más detallado
        JOptionPane.showMessageDialog(null, "Error inesperado: " + e.getMessage());
        // Considerar registrar la excepción
    }
}

```

Segunda pantalla del Código Depósito: Se muestra como se realiza la conexión a la base de datos donde se requiere el incremento del valor de saldo, también se muestra un serie de mensajes como el uso incorrecto de número decimales y negativos. mensajes de error y así por último el mensaje de depósito exitoso.

Captura de pantalla del código de Retiro.

```

import javax.swing.ImageIcon;
import java.awt.Color;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import javax.swing.JOptionPane;

/**
 *
 * @author ramon.valdez
 */
public class Retiro_JF extends javax.swing.JFrame {

    /**
     * Creates new form Retiro_JF
     */
    public Retiro_JF() {
        initComponents();
        setIconImage( new ImageIcon(getClass().getResource("/peso.PNG")).getImage());
        getContentPane().setBackground(new Color(217, 242, 72));
    }
}

```

Código de pantalla de Retiro: Solo mostramos un pequeño código que muestra las librerías utilizadas, la línea donde agregamos un icono en la parte de arriba de la aplicación al ejecutar la app y la línea de cambio de color de la interfaz.

```

private void BtnRetActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        Connection cnc = new Conexion().getConnection();
        PreparedStatement ps = cnc.prepareStatement("UPDATE cuenta SET saldo = saldo - ? WHERE id = 1");

        // Validar el valor ingresado
        String retiroStr = CtxtRet.getText();
        int Retiro;
        try {
            Retiro = Integer.parseInt(retiroStr);
            if (Retiro <= 0) {
                throw new NumberFormatException("El valor debe ser un número entero positivo.");
            }
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(null, "Por favor, ingrese un valor numérico entero positivo.");
            return; // Salir del método si no es un entero positivo
        }

        ps.setInt(1, Retiro);

        int rowsAffected = ps.executeUpdate();

        if (rowsAffected > 0) {
            JOptionPane.showMessageDialog(null, "Retiro exitoso.");
        } else {
            JOptionPane.showMessageDialog(null, "Error al realizar el retiro.");
        }

        CtxtRet.setText("");
        cnc.close();
    } catch (SQLException e) {
        // Manejo de excepciones más detallado
        JOptionPane.showMessageDialog(null, "Error inesperado: " + e.getMessage());
        // Considerar registrar la excepción
    }
}

```

Segunda pantalla Código de Retiro: Se muestra como se realiza la conexión a la base de datos

donde se requiere el incremento del valor de saldo, también se muestra un serie de mensajes como el uso incorrecto de número decimales y negativos. mensajes de error y así por último el mensaje de retiro exitoso

Captura de pantalla del código de Saldo.

```
import java.awt.Color;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;

/**
 *
 * @author ramon.valdez
 */
public class Saldo_JF extends javax.swing.JFrame {

    /**
     * Creates new form Saldo_JF
     */
    public Saldo_JF() {
        initComponents();
        setIconImage( new ImageIcon(getClass().getResource("/BCO.PNG")).getImage());
        getContentPane().setBackground(new Color(217, 242, 72));

        try (Connection cnx = new Conexion().getConnection();
            PreparedStatement ps = cnx.prepareStatement("SELECT saldo FROM cuenta WHERE id = 1");
            ResultSet rst = ps.executeQuery()) {

            if (rst.next()) {
                CtxtSal.setText(rst.getString("saldo"));
            } else {
                JOptionPane.showMessageDialog(null, "Error al mostrar el Saldo.");
            }
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(null, "Error en la base de datos: " + e.getMessage());
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(null, "Error al formatear el saldo.");
        }
    }

    private void BtnnSalActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.dispose();
    }
}
```

Código de pantalla de Saldo: Mostramos las librerías, la consulta de la conexión a la base de datos para obtener el saldo actual, así como también el código del botón salir de la ventana de la aplicación.

Captura de pantalla de las clases.

```

import java.sql.Connection;
import java.sql.DriverManager;

/**
 *
 * @author ramon.valdez
 */
public class Conexion {
    public Connection getConnection(){
        Connection cnc = null;
        String url = "jdbc:mysql://localhost:3306/bco_mxn";
        String user = "root";
        String pass = "4Pp.Bc0mX_A3";

        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            cnc = (Connection) DriverManager.getConnection(url, user, pass);
        } catch (Exception e) {
            System.out.println(e);
        }
        return cnc;
    }
}

```

Código de pantalla clase: Mostramos la clase de conexión la base de datos a utilizar en la aplicación del banco mexicano.

```

package com.mycompany.a2_banco_mxn;

/**
 *
 * @author ramon.valdez
 */
public class img_ico {
    public static void main(String[] args) {
        Menu_JF f = new Menu_JF();
        f.setVisible(true);
    }
}

```

Código de pantalla clase: Mostramos la clase de icono mostrados en las pantallas de la aplicación del banco mexicano.

Conclusion.

En conclusión: Esta segunda fase de conexión de la interfaz con la base de datos de aplicación causó un poco de confusión por algunos errores: uno presentado al momento de querer ejecutar la aplicación mandaba una incompatibilidad con la librería utilizada la cual se tuvo que modificar y usar una librería compatible. dos al ejecutar la aplicación enviaba un error de certificado ssl el cual se tuvo que desactivar desde la conexión del manejador de la base de datos de MySQL para que no le metiera ruido a la aplicación. tres al ingresar a los menús de depósito, retiro y saldo mostraba la ventana de error en cada una de las opciones, detectando el problema que el campo saldo no se encontraba con ningún monto asignado, con las situaciones presentadas se llegó a realizar aplicación de manera correcta dejando experiencia para nuevas situaciones en el futuro. La forma de realizar la app y la sintaxis del programa que se realizó en la herramientas utilizadas espero sea del agrado del docente de la materia para poder obtener la calificación y continuar con la actividad siguiente.

Referencias.

GitHub: Let's build from here. (n.d.)

Microsoft Copilot en Edge. (n.d.). Microsoft.com. Retrieved June 3, 2024, from

<https://www.microsoft.com/es-mx/edge/copilot?form=MT00IR&pl=launch>

No title. (n.d.). Chatgpt.com. Retrieved June 4, 2024, from <https://chatgpt.com/>