Universidad de San Carlos de Guatemala

Centro Universitario de Occidente

División de Ciencias de la Ingeniería

Área Profesional

Introducción a la Programación y Computación 2

Sección “A”

Ing. José Moisés Granados Guevara

“MANUAL TECNICO PRACTICA 1”

Melvin Eduardo Ordoñez Sapón RA | 202230552

Quetzaltenango, agosto del 2024

“Id y enseñad a todos”

ACERCA DEL PROGRAMA:

IDE utilizado: Apache NetBeans IDE 22

JDK utilizado: Versión 21.0.4

Programado en Sistema Operativo: Ubuntu 24.04 LTS

Terminal Utilizada: Terminal de Linux

Lenguaje de Programación Utilizado: Java

Base de datos utilizado: MySQL 24.04.1

Compatibilidad: Windows, Linux, macOS

METODOS IMPORTANTES

Este metodo es importante ya que es el que se utiliza para establecer la conexión a la base de datos, al utilizar esta aplicación el usuario debera de cambiar el usuario y contraseña para poder acceder a ella, antes de eso debera de crear la base de datos con el mapeo

```
public static Connection conectar()  throws SQLException {
   try {
      connection = DriverManager.getConnection(URL_MYSQL, USER, PASSWORD);
     //System.out.println("Conexión a la base de datos exitosa");
      return connection;
   } catch (SQLException ex) {
      ex.printStackTrace();
      throw new SQLException("Error al conectar con la base de datos", ex);
   }
}
```

Con este metodo se crea una solicitud nueva en la base de datos, para ello se realizan algunas validaciones antes de realizar esta accion

```
public void crearSolicitud(Solicitud solicitud) {
   String insert = "INSERT INTO solicitudes (numero_solicitud, fecha, tipo_tarjeta,
nombre_solicitante, salario, direccion) VALUES " +
         "('" + solicitud.getNumeroSolicitud() + "', '" +
Herramientas.formatoFecha(solicitud.getFecha()) + "', '" + solicitud.getTipo() + "', '" +
solicitud.getNombreSolicitante() + "', " +
         solicitud.getSalario() + ", '" + solicitud.getDireccion() + "')";

   try {
```

```
        Statement statementInsert = connection.createStatement();
        int rowsAffected = statementInsert.executeUpdate(insert);
        System.out.println("Rows affected> " + rowsAffected);
        JOptionPane.showMessageDialog(null, "Solicitud Registrada");

    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Error al insertar en la base de datos");
        System.out.println("Error al insertar en la base de datos");
    }
}
```

Con este metodo se realizan las respectivas validaciones para ver si se debe de registrar una tarjeta o no

```
public void crearTarjeta(Tarjeta tarjeta) {

    //TipoTarjeta tipoTarjeta, String nombreTitular, String direccion, String fechaSolicitud)
    String insert = "INSERT INTO tarjetas (numero_tarjeta, tipo, limite, nombre, direccion, fecha,
numero_solicitud) VALUES "
            + "('" + tarjeta.getNumeroTarjeta() + "', '" + tarjeta.getTipoTarjeta() + "', " +
tarjeta.getLimite() + ", '" + tarjeta.getNombreTitular() + "', '"
            + tarjeta.getDireccion() + "', '" + tarjeta.getFechaSolicitud() + "', '" +
tarjeta.getNumeroSolicitud() + "')";

    try {
        Statement statementInsert = connection.createStatement();
        int rowsAffected = statementInsert.executeUpdate(insert);
        System.out.println("Tarjeta agregada");
        JOptionPane.showMessageDialog(null, "Tarjeta agregada");

    } catch (SQLException e) {
        System.out.println("Error al insertar en la tabla tarjetas");
        JOptionPane.showMessageDialog(null, "Error al insertar en la tabla tarjetas");
    }
}
```

Para registrar un nuevo movimiento se utiliza el siguiente método donde antes de, se realizan las validaciones correspondientes para evitar errores de ejecución

```
public void registrarMovimiento(MovimientoTarjeta movimientoTarjeta) {
    //Validar si la tarjeta está activa
    ValidarMovimientoDB validarMovimiento = new ValidarMovimientoDB();
    if (!validarMovimiento.estaActivo(movimientoTarjeta)) {
        return;
    }

    //Validar le alcanza el credito
    if (!validarMovimiento.tieneCreditoSuficiente(movimientoTarjeta)) {
        return;
    }

    String insert = "INSERT INTO movimientos (numero_tarjeta, fecha, tipo_movimiento,
descripcion, establecimiento, monto) VALUES " +
            "('" + movimientoTarjeta.getNumeroTarjeta() + "', '" +
Herramientas.formatoFecha(movimientoTarjeta.getFecha()) + "', '" +
movimientoTarjeta.getTipoMovimiento() + "', '" + movimientoTarjeta.getDescripcion() + "', '" +
movimientoTarjeta.getEstablecimiento() + "', " +
            movimientoTarjeta.getMonto() + ")";

    try {
        Statement statementInsert = connection.createStatement();
        int rowsAffected = statementInsert.executeUpdate(insert);
        System.out.println("Movimiento registrado");
        JOptionPane.showMessageDialog(null, "Movimiento registrado");

    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Error al agregar movimiento");
        System.out.println("Error al agregar movimiento");
    }
}
```

Con este metodo se calcula cual es el interés que debe de ser aplicado al propietario de la tarjeta en base al tipo que posee

```
public static double establecerInteres(String numeroTarjeta)  {
  {
     try{
         connection = ConexionMySQL.conectar();
     }
     catch (SQLException e) {
        System.out.println("Error al conectar a la base de datos: " + e.getMessage());
     }

     String tipoTarjeta = null;
     double interes = 0.0;

     // Consulta para obtener el tipo de tarjeta
     String consulta = "SELECT tipo FROM tarjetas WHERE numero_tarjeta = ?";

     try {
        // Preparar la consulta
        PreparedStatement statementConsulta = connection.prepareStatement(consulta);
        statementConsulta.setString(1, numeroTarjeta);
        ResultSet resultSet = statementConsulta.executeQuery();

        // Verificar si se encontró la tarjeta
        if (resultSet.next()) {
           tipoTarjeta = resultSet.getString("tipo");

           // Calcular el interés basado en el tipo de tarjeta
           switch (tipoTarjeta) {
              case "NACIONAL":
                 interes = 0.012; // 1.2%
                 break;
              case "REGIONAL":
                 interes = 0.023; // 2.3%
                 break;
              case "INTERNACIONAL":
                 interes = 0.0375; // 3.75%
                 break;
              default:
                 System.out.println("Tipo de tarjeta desconocido");
                 break;
```

```
            }
        } else {
            JOptionPane.showMessageDialog(null, "Tarjeta no encontrada");
            System.out.println("Tarjeta no encontrada");
        }
    } catch (SQLException e) {
        System.out.println("Error al obtener el tipo de tarjeta: ");
    }
    return interes;
  }
}
```

Este método se utiliza en algunas operaciones para formatear el formato de la fecha a uno compatible con la base de datos

```
public static String formatoFecha(String fecha) {
  fecha = fecha.trim();
  String[] fechaArray = fecha.split("/");
  String fechaFormateada = fechaArray[2] + "-" + fechaArray[1] + "-" + fechaArray[0];
  return fechaFormateada;
}
```