UCA
UNIVERSIDAD
CENTROAMERICANA

Facultad de
Ciencia, Tecnología y Ambiente

Ingeniería en Sistemas de Información
Programación Orientada a Objetos

CASO DE ESTUDIO
Trabajo final

Jason De Jesús Esquivel Rivera

Leoncio José Pavón Larios

Jan Carlos Prado Mayorga

Docente:

Prof. Ing. José Alejandro Durán G.

Descripción del caso de estudio

Se contactó con una distribuidora de GLP (gas licuado de petróleo), se realizó una reunión con los propietarios de dicho negocio, se mantuvo una charla en la cual nos presentaron la siguiente deficiencia que lograron detectar dentro de su negocio: la necesidad de un registro de asistencia de los trabajadores, debido a que al ser un numero de sedes diferentes, no se podía llevar un control único de asistencia, debido a que era mucho trabajo movilizar la asistencia de varios trabajadores sin ID a las distintas sedes para llevar un control regular.

Se nos solicitó la creación de un algoritmo mediante el cual se accediera a la información de un trabajador mediante el número de ID y que registrara la información recopilada de todos los trabajadores en las distintas sedes de la empresa. Se nos solicitó la hora y fecha de entrada y salida, un inicio de sesión para poder regular quienes acceden al sistema y un acceso a la plantilla de asistencia.

Tabla de contenido

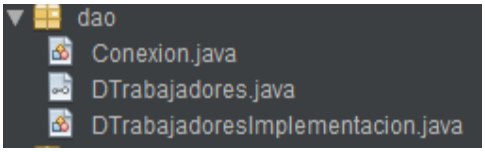
1. Explicación del Programa de asistencia

El programa fue creado en el lenguaje Java, con el patrón de diseño Dao (Data Access Object) que consiste en separar la lógica de acceso de datos de los Objetos, luego de esto creamos las clases correspondientes que son la “Conexión”, “Trabajadores” y “DTrabajadoresImplementacion”.

```java
public class Conexion {

    private static Conexion instancia;
    private static final String SERVIDOR = "localhost";
    private static final String USUARIO = "sa";
    private static final String PW = "1234";
    private static final String NOMBREBD = "Proyecto_SystemColor";
    private static final String PUERTO = "1434";
    private static final String DRIVER = "com.microsoft.sqlserver.jdbc.SQLServerDriver";
```

En la clase Conexión tenemos diferentes variables para la conexión con SQL SERVER, en la siguiente llamada Trabajadores tenemos los métodos que se usaron en la siguiente clase que lleva por nombre DTrabajadoresImplementacion, en está se les agrego la conexión, la petición a SQL y se le mandaron los respectivos datos a Agregar, Modificar, Eliminar y Buscar ya que nuestro programa se baso en un CRUD (Create, Read, Update, Delete).




```java
public interface DTrabajadores {

    public void registrar(Trabajadores trabajadores);

    public void modificar(Trabajadores trabajadores);

    public void eliminar(Trabajadores trabajadores);

    public void buscar(Trabajadores trabajadores);

}
```

Luego creamos un paquete donde están las 4 interfaces utilizadas en este programa las cuales son, FrmPrincipal que cuenta que con un solo botón que nos manda a la siguiente interfaz llamada FrmLogin acá dejamos definidos los datos de ingreso ya que los trabajadores solo entrar a registrar su entrada más nada, igualmente está validado para cualquier error.

```java
public void datos(String us, String pas) {
    usuario = "Admin";
    contraseña = "1234";
}
```

Luego de que iniciemos sesión, nos mandara a la interfaz FrmRegistro, acá viene la magia, se piden los datos al usuario y tienen diferentes funciones a la mano, como las mencionadas anteriormente guardar, eliminar, buscar y modificar, todas validadas para evitar problemas con los datos mandados a SQL.

```
private void btnBuscarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    DTrabajadoresImplementacion trabajadores_dao = new DTrabajadoresImplementacion();
    Trabajadores trabajador = new Trabajadores();

    if (txtClave.getText().equals("")) {
        JOptionPane.showMessageDialog(null, "Ingrese la Clave de Registro a Buscar", "Aviso Gas Butano S.A.", 1);
    } else {
        int key = Integer.parseInt(txtClave.getText());
        trabajador.setClave(key);
        trabajadores_dao.buscar(trabajador);

        txtClave.setText(String.valueOf(trabajador.getClave()));
        txtNombre.setText(trabajador.getNombre());
        txtDepartamento.setText(trabajador.getDepartamento());
        txtTelefono.setText(trabajador.getTelefono());
        txtFecha_entrada.setText(trabajador.getFecha_entrada());
        txtHora_entrada.setText(trabajador.getHora_entrada());
        txtCargo.setText(trabajador.getCargo());

        JOptionPane.showMessageDialog(null, "Registro Encontrado", "Aviso Gas Butano S.A.", 1);
    }

}
```

Esa seria una de las implementaciones, primero se crean los objetos, luego las condiciones, ya que pase la primera condición se crean variables para recibir los datos ingresados y luego se mandan/piden a SQL mediante los objetos creados y se muestra un mensaje con la frase Registro encontrado.

Igualmente tenemos un botón creado para mostrar la tabla donde están todos los registros, en esta existe un botón que manda una petición a SQL y este le regresa los datos ya ingresados anteriormente por el usuario.

```
public void mostrar(String tabla) {
    String sql = "SELECT * FROM " + tabla;
    Statement st;
    Conexion conn = new Conexion();
    Connection conexion = conn.getConexion();
    DefaultTableModel model = new DefaultTableModel();
    model.addColumn("clave");
    model.addColumn("nombre");
    model.addColumn("departamento");
    model.addColumn("telefono");
    model.addColumn("fecha_entrada");
    model.addColumn("hora_entrada");
    model.addColumn("cargo");
    tablaPersonal.setModel(model);

    String[] datos = new String[7];
    try {
        st = conexion.createStatement();
        ResultSet rs = st.executeQuery(sql);
        while (rs.next()) {
            datos[0] = rs.getString(1);
            datos[1] = rs.getString(2);
            datos[2] = rs.getString(3);
            datos[3] = rs.getString(4);
            datos[4] = rs.getString(5);
            datos[5] = rs.getString(6);
            datos[6] = rs.getString(7);
            model.addRow(datos);
        }
        JOptionPane.showMessageDialog(null, "Registros Encontrados y/o No Encontrados", "Aviso Gas Butano S.A.", 1);
        conexion.close();
    } catch (SQLException ex) {
        System.out.println(ex);
    }
}
```

Para finalizar agregamos validaciones en todas las interfaces, menos en la Principal ya que solo es un botón, igualmente tenemos botones para cerrar el programa con la función System.exit(0).

2. Requerimientos del programa

El negocio pretende instalar el programa **registro de entrada de gas butano S.A.** en una de las distintas sedes que posee la empresa. Para permitir a los gerentes del local acceder a la información se les entregara un usuario y contraseña por defecto (Figura 1.1). el gerente debe iniciar sesión para acceder a la iniciación del registro (es decir, ingresar la fecha hora e ID de los empleados)
La interfaz del usuario contiene los siguientes componentes

- Una pantalla que exige un nombre de usuario y contraseña
- Un botón para iniciar sesión
- Un botón para poder salir del login (aparte del botón cerrar en la parte superior derecha)

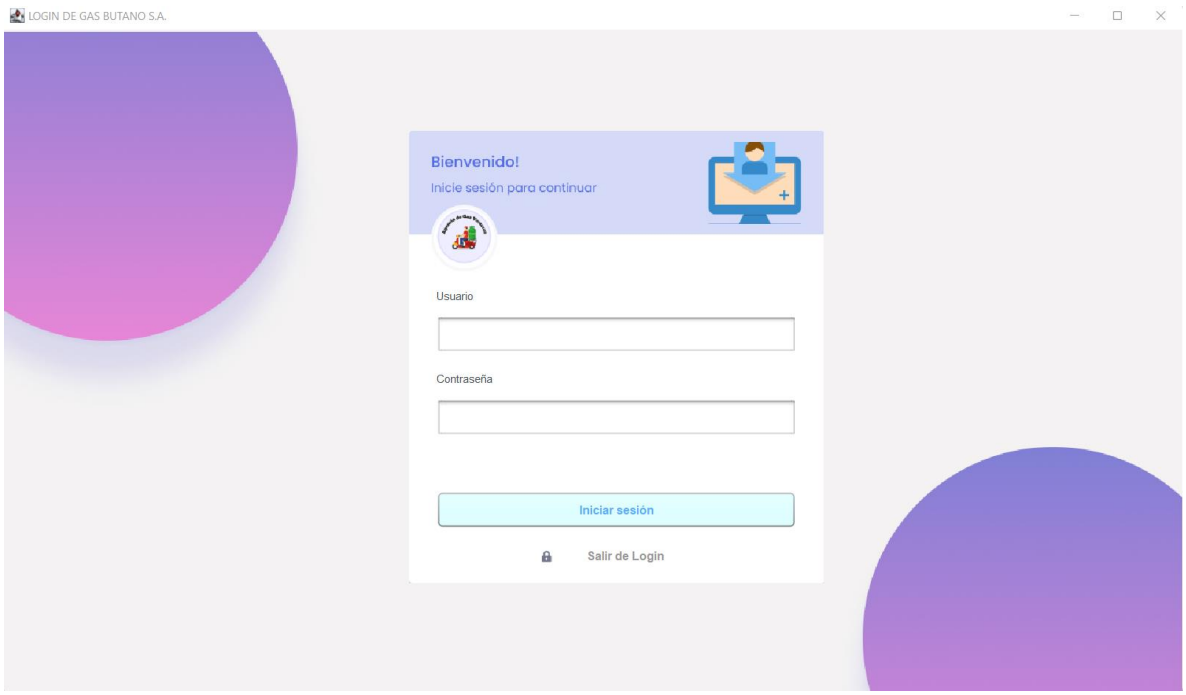


Figura 1.1 – inicio de sesión en el programa dado.

El inicio de sesión del programa consiste en la validación del usuario y contraseña con respecto a la escrita principalmente en el código, es decir el programa evalúa si la contraseña es correcta relacionada con la escrita en el programa, también evalúa si el usuario es el mismo (diferenciando entre mayúsculas y minúsculas en ambos casos) si ambos campos son completamente correctos el programa iniciaría sesión, en cambio sí solo la contraseña o solo el usuario es correcto el programa enviaría una notificación explicando que uno de los dos campos es incorrecto (Figura 1.2)

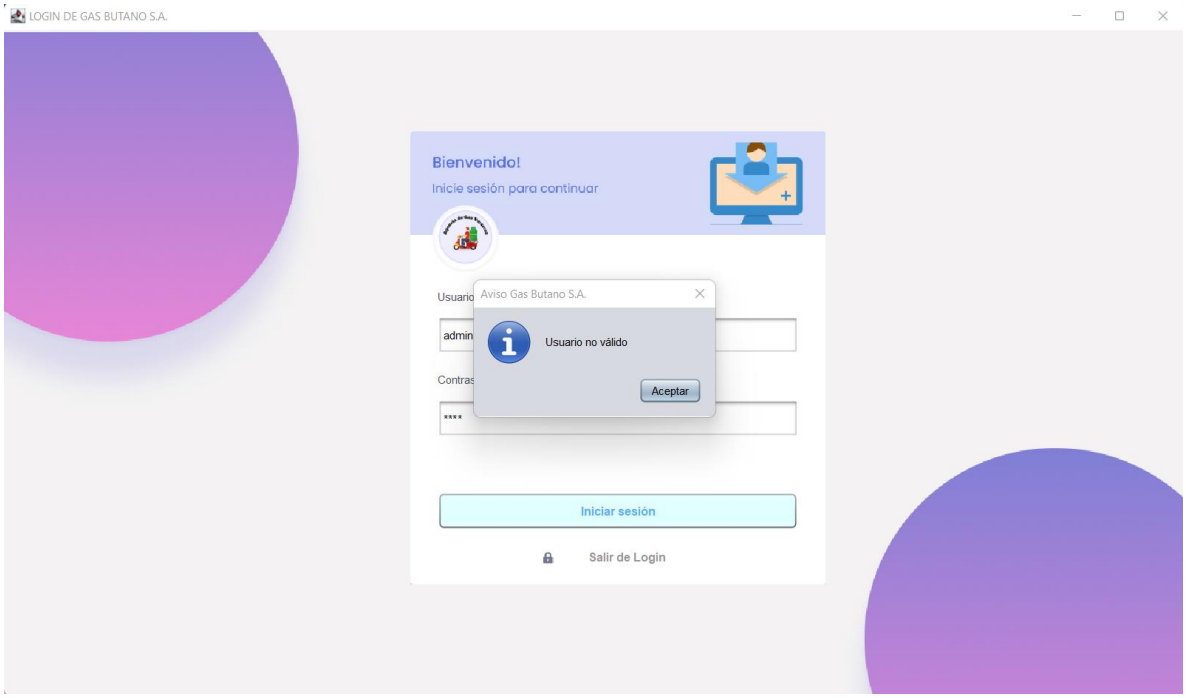


Figura 1.2 – inicio de sesión fallido.

3. después de la autenticación.

Una vez el programa ha comprueba que tanto la contraseña como el usuario son correctos se mostrará el menú de registro (Figura 1.3) en el cual se podrá ingresar los campos de clave, nombre, departamento, teléfono, la fecha y la hora de entrada y salida.

Después de haberse llenado la información se pueden seguir las siguientes interacciones. Un botón para limpiar el programa de registro en pantalla, un botón para buscar los trabajadores según el número de ID, un botón para guardar la información de la asistencia de un trabajador, un botón para eliminar al trabajador y otro en el que se puede ver la hoja de asistencia en tiempo real (Figura 1.4).

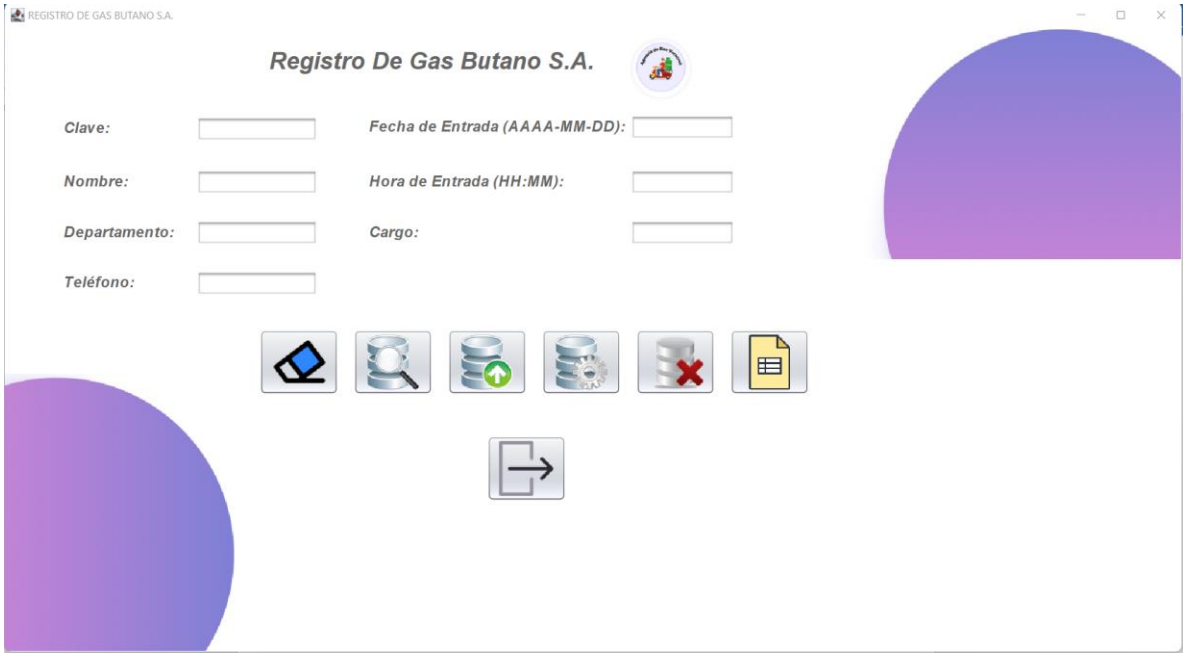


Figura 1.3 –Menú de registro.

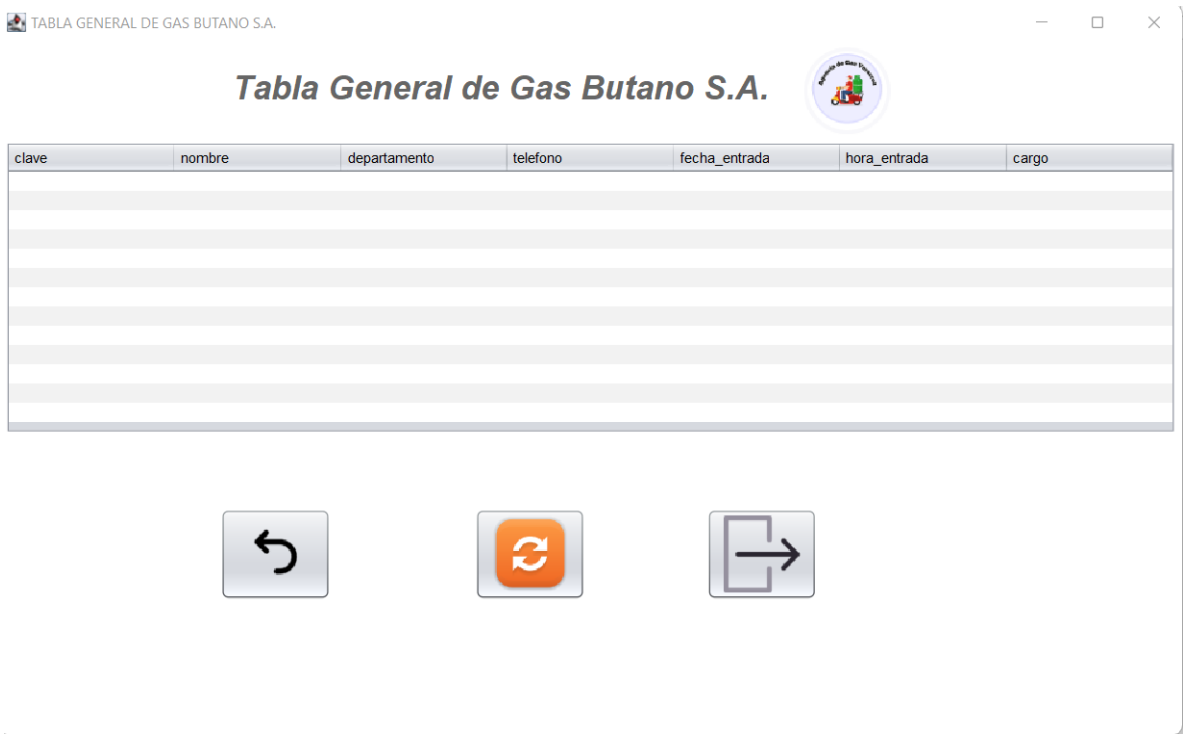


Figura 1.4 – tabla de asistencia.

en la tabla de asistencia podemos ver el registro de los trabajadores que han entrado, salido, la fecha y la hora, esto sirve para que el gerente pueda ver el ingreso diario sin necesidad de realizar la búsqueda en el botón buscar, de esta manera es más fácil de comprenderse y de llevar el control, en la pantalla del registro de asistencia de pueden contemplar las siguientes opciones (Figura 1.4) un

botón para devolverse al menú principal, un botón para refrescar la hoja de asistencia y otro para cerrar sesión directamente.

4. modelado de las clases

Se crean dos clases el login y los trabajadores (figura 1.5) en estas se relacionan con una dependencia, debido a que el acceso al ingreso de los trabajadores depende del inicio de sesión, solo existen 3 puestos de trabajadores (el gerente puede contactarse con nosotros para agregar más puestos a su sucursal) los repartidores, cajeros y camioneros.

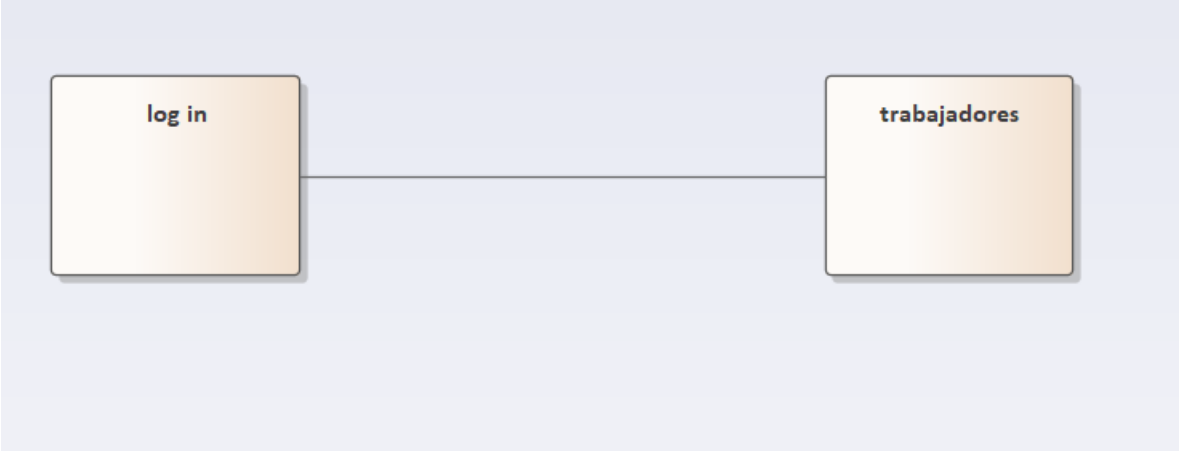


Figura 1.5 –clases creadas.

En las clases se puede ver que el login tiene dos variables (figura 1.6) el usuario y la contraseña, estas son declaradas como string (debido a que es posible traducir números a letras con una función en el programa) mientras que los trabajadores son creados con más variables, (todas las mencionadas en la parte de registro) y con los atributos respectivos (figura 1.7)

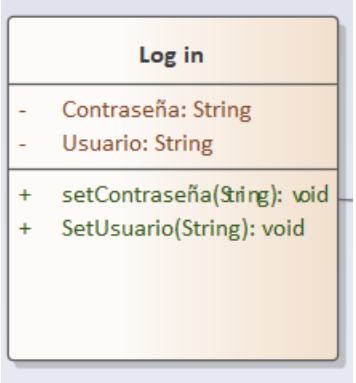


Figura 1.6 – variables en la clase log in

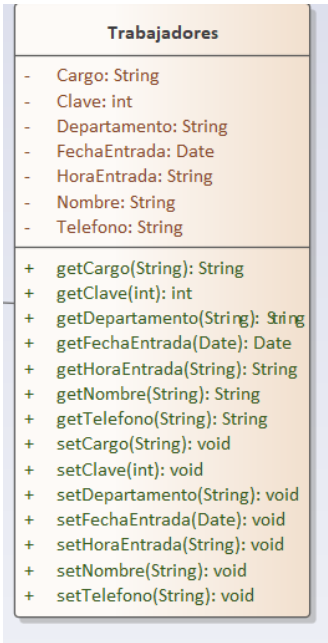


Figura 1.7 –variables en la clase trabajadores.

El programada creado puede ayudar a la verificación de entrada de los trabajadores en las distintas sedes de gas butano S.A. este programa fue creado con la intención de crear una facilidad a la hora de trabajar con una empresa grande, con distintos puestos de trabajo, se examinaron la mayor cantidad de variables a la toma de asistencia de un trabajador y utilidades en el programa para poder tener más utilidades, en caso de cometerse errores por parte del empleador, un ingreso tardío accidental puede ser modificado y una asistencia inexistente puede ser eliminada.