

## Registro

Nos permite agregar una nueva persona para que pueda ingresar al sistema esto solo lo puede hacer el Administrador.



```
public class Registro extends javax.swing.JFrame {
    login lg = new login();
    LoginDAO login = new LoginDAO();

    public Registro() {
        initComponents();
        this.setLocationRelativeTo(null);
    }

    public void validar(){
        String correo = txtCorreo.getText();
        String pass = String.valueOf(txtPass.getPassword());
        String nom = txtNombre.getText();
        String rol = cbxRol.getSelectedItem().toString();
        if (!"".equals(correo) && !"".equals(pass) && !"".equals(nom)) {
            lg.setNombre(nom);
            lg.setCorreo(correo);
            lg.setPass(pass);
            lg.setRol(rol);
            login.Registrar(lg);
            lg = login.log(correo, pass);
            Login iniciar = new Login();
            iniciar.setVisible(true);
            dispose();
        }
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code
```

## Login



The image shows a login form for 'Farmacia Bethesda'. On the left, there is a logo featuring a blue cross inside a circle with a hand-like shape, and the text 'Farmacia Bethesda' below it. On the right, there is a white panel with a blue border. At the top of this panel is a blue padlock icon. Below the icon, there are three input fields: one for 'Correo electrónico', one for 'Contraseña', and a larger one for a button. The labels 'Correo electrónico' and 'Contraseña' are in blue text.

Aca estan los botones que se utilizan al iniciar el login.

```

public class Login extends javax.swing.JFrame {
    login lg = new login();
    LoginDAO login = new LoginDAO();

    public Login() {
        initComponents();
        this.setLocationRelativeTo(null);
    }

    public void validar(){
        String correo = txtCorreo.getText();
        String pass = String.valueOf(txtPass.getPassword());
        if (!"".equals(correo) || !"".equals(pass)) {

            lg = login.log(correo, pass);
            if (lg.getCorreo() != null && lg.getPass() != null) {
                Sistema sis = new Sistema(lg);
                sis.setVisible(true);
                dispose();
            }
            else{
                JOptionPane.showMessageDialog(null, "correo o contraseña incorrecta");
            }
        }
    }
}

```

A

```

L  */
public class login {
    private int id;
    private String nombre;
    private String correo;
    private String pass;
    private String rol;

    public login() {
    }

    public login(int id, String nombre, String correo, String pass, String rol) {
        this.id = id;
        this.nombre = nombre;
        this.correo = correo;
        this.pass = pass;
        this.rol = rol;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getCorreo() {
        return correo;
    }
}

public class LoginDAO {
    Connection con;
    PreparedStatement ps;
    ResultSet rs;
    Conexion cn = new Conexion();

    public login log(String correo, String pass){
        login l = new login();
        String sql = "SELECT * FROM usuarios WHERE correo = ? AND pass = ?";
        try{
            con = cn.getConnection();
            ps = con.prepareStatement(sql);
            ps.setString(1, correo);
            ps.setString(2, pass);
            rs = ps.executeQuery();
            if (rs.next()){
                l.setId(rs.getInt("Id"));
                l.setNombre(rs.getString("nombre"));
                l.setCorreo(rs.getString("correo"));
                l.setPass(rs.getString("pass"));
                l.setRol(rs.getString("rol"));
            }
        }catch (SQLException e){
            System.out.println(e.toString());
        }
        return l;
    }

    public boolean Registrar(login reg){
        String sql = "INSERT INTO usuarios (nombre, correo, pass, rol) VALUES (?, ?, ?, ?)";
        try{
            con = cn.getConnection();
            ps = con.prepareStatement(sql);
            ps.setString(1, reg.getNombre());
            ps.setString(2, reg.getCorreo());
            ps.setString(3, reg.getPass());
            ps.setString(4, reg.getRol());
            ps.execute();
            return true;
        }catch (SQLException e){
            System.out.println(e.toString());
            return false;
        }
    }
}

```

La venta principal contiene muchos botones los cuales están asignados para acción que queramos realizar dejare una breve explicación de los botones que usa cada modulo

## Ventas Nuevas

**Punto de Venta**

Nueva Venta | Clientes | Proveedor | Productos | Ventas | Config

Selecccionar

Código Descripción Cantidad Precio Stock Disponible

CODIGO	DESCRIPCION	CANTIDAD	PRECIO U.	TOTAL
--------	-------------	----------	-----------	-------

DPI/RUC NOMBRE

Total a Pagar

```
private void btnNuevaVentaActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    jTabbedPanel.setSelectedIndex(0);  
}  
  
private void txtCodigoVentaActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
}  
  
private void btnEliminarventaActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    modelo = (DefaultTableModel) TableVenta.getModel();  
    modelo.removeRow(TableVenta.getSelectedRow());  
    TotalPagar();  
    txtCodigoVenta.requestFocus();  
}
```

```

private void txtCodigoVentaKeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    if(evt.getKeyCode() == KeyEvent.VK_ENTER){
        if(!"".equals(txtCodigoVenta.getText())){
            String cod = txtCodigoVenta.getText();
            pro = proDao.BuscarPro(cod);
            if(pro.getNombre() != null){
                txtDescripcionVenta.setText(""+pro.getNombre());
                txtPrecioVenta.setText(""+pro.getPrecio());
                txtStockDisponible.setText(""+pro.getStock());
                txtCantidadVenta.requestFocus();
            }else{
                JOptionPane.showMessageDialog(null, "El codigo no existe");
                LimpiarVenta();
                txtCodigoVenta.requestFocus();
            }
        }else{
            JOptionPane.showMessageDialog(null, "Ingrese el codigo del Producto");
            txtCodigoVenta.requestFocus();
        }
    }
}

private void txtCantidadVentaKeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    if(evt.getKeyCode() == KeyEvent.VK_ENTER){
        if(!"".equals(txtCantidadVenta.getText())){
            String cod = txtCodigoVenta.getText();
            String descripcion = txtDescripcionVenta.getText();
            int cant = Integer.parseInt(txtCantidadVenta.getText());
            double precio = Double.parseDouble(txtPrecioVenta.getText());
            double total = cant * precio;
            int stock = Integer.parseInt(txtStockDisponible.getText());
            if(stock >= cant){
                item = item + 1;
                tmp = (DefaultTableModel) TableVenta.getModel();
                for(int i = 0; i < TableVenta.getRowCount(); i++){
                    if(TableVenta.getValueAt(i, 1).equals(txtDescripcionVenta.getText())){
                        JOptionPane.showMessageDialog(null, "El producto ya esta registrado");
                        return;
                    }
                }
            }
        }
    }
}

```

## Cientes

The screenshot shows a Java Swing window titled 'Clientes'. On the left is a sidebar with six buttons: 'Nueva Venta' (with a document icon), 'Clientes' (with a group of people icon), 'Proveedor' (with a person icon), 'Productos' (with a book icon), 'Ventas' (with a shopping cart icon), and 'Config' (with a gear icon). The main content area is divided into two parts. The top part is a form with five labeled text input fields: 'DPI/RUC:', 'Nombre:', 'Telefono:', 'Dirección:', and 'Razón Social:'. Below these fields are four square buttons with icons: a green arrow pointing down, a blue circle with a white 'A', a red 'X', and a green circle with a white plus sign. The bottom part of the main area is a table with the following columns: 'ID', 'DPI', 'NOMBRE', 'TELEFONO', 'DIRECCION', and 'RAZON SOCIAL'. The table is currently empty.

Aca estan los codigos de los botones utilizados en la ventana clientes

```

private void btnGuardarClienteActionPerformed(java.awt.event.ActionEvent evt) {
    if (!"".equals(txtRucConfig.getText()) && !"".equals(txtNombreConfig.getText()) && !"".equals(txtTelefonoConfig.getText()) && !"".equals(txtDireccionConfig.getText()) &&
        conf.setRuc(Integer.parseInt(txtRucConfig.getText()));
        conf.setNombre(txtNombreConfig.getText());
        conf.setTelefono(Integer.parseInt(txtTelefonoConfig.getText()));
        conf.setDireccion(txtDireccionConfig.getText());
        conf.setRazon(txtRazonConfig.getText());
        conf.setId(Integer.parseInt(txtIdConfig.getText()));
        proDao.ModificarDatos(conf);
        JOptionPane.showMessageDialog(null, "Cliente Registrado");
        ListarConfig();
    } else {
        JOptionPane.showMessageDialog(null, "Los campos están vacíos");
    }
}

```

```

private void btnClientesActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    LimpiarTable();
    ListarCliente();
    jTable1.setSelectedIndex(1);
}

```

```

private void TableClienteMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int fila = TableCliente.rowAtPoint(evt.getPoint());
    txtIdCliente.setText(TableCliente.getValueAt(fila, 0).toString());
    txtDniCliente.setText(TableCliente.getValueAt(fila, 1).toString());
    txtNombreCliente.setText(TableCliente.getValueAt(fila, 2).toString());
    txtTelefonoCliente.setText(TableCliente.getValueAt(fila, 3).toString());
    txtDireccionCliente.setText(TableCliente.getValueAt(fila, 4).toString());
    txtRazonCliente.setText(TableCliente.getValueAt(fila, 5).toString());
}

```

```

private void btnEliminarClienteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if (!"".equals(txtIdCliente.getText())) {
        int pregunta = JOptionPane.showConfirmDialog(null, "Estas seguro de eliminar");
        if (pregunta == 0) {
            int id = Integer.parseInt(txtIdCliente.getText());
            client.EliminarCliente(id);
            LimpiarTable();
            LimpiarCliente();
            ListarCliente();
        }
    }
}

```

```

public boolean RegistrarCliente(Cliente cl){
    String sql = "INSERT INTO clientes (dni, nombre, telefono, direccion, razon) VALUES (?, ?, ?, ?, ?)";
    try{
        con = cn.getConnection();
        ps = con.prepareStatement(sql);
        ps.setInt(1, cl.getDni());
        ps.setString(2, cl.getNombre());
        ps.setInt(3, cl.getTelefono());
        ps.setString(4, cl.getDireccion());
        ps.setString(5, cl.getRazon());
        ps.execute();
        return true;
    }catch (SQLException e){
        JOptionPane.showMessageDialog(null, e.toString());
        return false;
    }
    finally{
        try{
            con.close();
        }catch (SQLException e){
            System.out.println(e.toString());
        }
    }
}

```



```

public class ProveedorDAO {
    Connection con;
    Conexion cn = new Conexion();
    PreparedStatement ps;
    ResultSet rs;

    public boolean RegistrarProveedor(Proveedor pr){
        String sql = "INSERT INTO proveedor(ruc, nombre, telefono, direccion, razon) VALUES (?, ?, ?, ?, ?)";
        try{
            con = cn.getConnection();
            ps = con.prepareStatement(sql);
            ps.setInt(1, pr.getRuc());
            ps.setString(2, pr.getNombre());
            ps.setInt(3, pr.getTelefono());
            ps.setString(4, pr.getDireccion());
            ps.setString(5, pr.getRazon());
            ps.execute();
            return true;
        }catch (SQLException e){
            System.out.println(e.toString());
            return false;
        }finally{
            try {
                con.close();
            } catch (SQLException e) {
                System.out.println(e.toString());
            }
        }
    }
}

```

```

public List ListarProveedor(){
    List<Proveedor> Listapr = new ArrayList();
    String sql = "SELECT * FROM proveedor";
    try{
        con = cn.getConnection();
        ps = con.prepareStatement(sql);
        rs = ps.executeQuery();
        while (rs.next()){
            Proveedor pr = new Proveedor();
            pr.setId(rs.getInt("id"));
            pr.setRuc(rs.getInt("ruc"));
            pr.setNombre(rs.getString("nombre"));
            pr.setTelefono(rs.getInt("telefono"));
            pr.setDireccion(rs.getString("direccion"));
            pr.setRazon(rs.getString("razon"));
            Listapr.add(pr);
        }
    }catch (SQLException e){
        System.out.println(e.toString());
    }
    return Listapr;
}

```

```

public boolean EliminarProveedor(int id){
    String sql = "DELETE FROM proveedor WHERE id=?";
    try{
        con = cn.getConnection();
        ps = con.prepareStatement(sql);
        ps.setInt(1, id);
        ps.execute();
        return true;
    }catch (SQLException e){
        System.out.println(e.toString());
        return false;
    }finally{
        try {
            con.close();
        } catch (SQLException e) {
            System.out.println(e.toString());
        }
    }
}

```



## Productos

Nueva Venta

Cientes

Proveedor

Productos

Ventas

Config

Código:

Descripción:

Cantidad:

Precio

Proveedor:











ID	CODIGO	DESCRIPCION	PROVEEDOR	CANTIDAD	PRECIO
----	--------	-------------	-----------	----------	--------

Aca estan los codigos de los botones utilizados en la ventana de productos

```
private void btnProductosActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    LimpiarTable();
    LimpiarProductos();
    ListarProductos();
    jTable1.setSelectedIndex(3);
}

```

```
private void TableProductoMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int fila = TableProducto.rowAtPoint(evt.getPoint());
    txtIdPro.setText(TableProducto.getValueAt(fila, 0).toString());
    txtCodigoPro.setText(TableProducto.getValueAt(fila, 1).toString());
    txtDesPro.setText(TableProducto.getValueAt(fila, 2).toString());
    cbxProveedorPro.setSelectedItem(TableProducto.getValueAt(fila, 3).toString());
    txtCantPro.setText(TableProducto.getValueAt(fila, 4).toString());
    txtPrecioPro.setText(TableProducto.getValueAt(fila, 5).toString());
}

```

```
private void btnEliminarProActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if (!"".equals(txtIdPro.getText())) {
        int pregunta = JOptionPane.showConfirmDialog(null, "Estas seguro de eliminar");
        if (pregunta == 0) {
            int id = Integer.parseInt(txtIdPro.getText());
            proDao.EliminarProductos(id);
            LimpiarTable();
            LimpiarProductos();
            ListarProductos();
        }
    }
}

```

```
public Config BuscarDatos(){
    Config conf = new Config();
    String sql = "SELECT * FROM config";
    try{
        con = cn.getConnection();
        ps = con.prepareStatement(sql);
        rs = ps.executeQuery();
        if(rs.next()){
            conf.setId(rs.getInt("id"));
            conf.setRuc(rs.getInt("ruc"));
            conf.setNombre(rs.getString("nombre"));
            conf.setTelefono(rs.getInt("telefono"));
            conf.setDireccion(rs.getString("direccion"));
            conf.setRazon(rs.getString("razon"));
        }
    }catch (SQLException e){
        System.out.println(e.toString());
    }
    return conf;
}

```

```
public boolean ModificarDatos(Config conf){
    String sql = "UPDATE config SET ruc=?, nombre=?, telefono=?, direccion=?, razon=? WHERE id=?";
    try{
        ps = con.prepareStatement(sql);
        ps.setInt(1, conf.getRuc());
        ps.setString(2, conf.getNombre());
        ps.setInt(3, conf.getTelefono());
        ps.setString(4, conf.getDireccion());
        ps.setString(5, conf.getRazon());
        ps.setInt(6, conf.getId());
        ps.execute();
        return true;
    }catch (SQLException e){
        System.out.println(e.toString());
        return false;
    } finally{
        try {
            con.close();
        } catch (SQLException ex) {
            System.out.println(ex.toString());
        }
    }
}

```

```

82 public Productos BuscarPorDescripcion(String descripcion) {
83     Productos producto = new Productos();
84     String sql = "SELECT * FROM productos WHERE nombre = ?";
85     try {
86         con = cn.getConnection(); // Obtener la conexión
87         ps = con.prepareStatement(sql);
88         ps.setString(1, descripcion);
89         rs = ps.executeQuery();
90         if (rs.next()) {
91             producto.setCodigo(rs.getString("codigo")); // Asumiendo que tienes un método setCodigo
92             producto.setNombre(rs.getString("nombre"));
93             producto.setPrecio(rs.getDouble("precio"));
94             producto.setStock(rs.getInt("stock"));
95         }
96     } catch (SQLException e) {
97         System.out.println(e.toString());
98     } finally {
99         try {
100             if (rs != null) rs.close();
101             if (ps != null) ps.close();
102             if (con != null) con.close();
103         } catch (SQLException ex) {
104             System.out.println(ex.toString());
105         }
106     }
107     return producto;
108 }
109
110 }
111
112

```

## Ventas

ID	CLIENTE	VENDEDOR	TOTAL
----	---------	----------	-------

```

private void btnGenerarVentaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if (TableVenta.getRowCount() > 0) {
        if (!"".equals(txtNombreClienteVenta.getText())) {
            RegistrarVenta();
            RegistrarDetalle();
            ActualizarStock();
            pdf();
            LimpiarTableVenta();
            LimpiarClienteVenta();
        } else {
            JOptionPane.showMessageDialog(null, "Debes ingresar un cliente");
        }
    } else {
        JOptionPane.showMessageDialog(null, "NO hay productos en la venta");
    }
}

```

## Config

Nueva Venta Clientes Proveedor Productos Ventas **Config**

### DATOS DE LA EMPRESA

RUC	NOMBRE	TELEFONO
<input type="text"/>	<input type="text"/>	<input type="text"/>
DIRECCION	RAZON SOCIAL	
<input type="text"/>	<input type="text"/>	

 ACTUALIZAR

```
public class Config {
    private int id;
    private int ruc;
    private String nombre;
    private int telefono;
    private String direccion;
    private String razon;

    public Config(){

    }

    public Config(int id, int ruc, String nombre, int telefono, String direccion, String razon) {
        this.id = id;
        this.ruc = ruc;
        this.nombre = nombre;
        this.telefono = telefono;
        this.direccion = direccion;
        this.razon = razon;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public int getRuc() {
        return ruc;
    }
}
```

## Base de datos

The image shows two screenshots related to a database setup on a Windows system.

**Top Screenshot: XAMPP Control Panel v3.3.0**

The XAMPP Control Panel shows the status of various services. The MySQL service is highlighted in green, indicating it is running. The log window shows the following messages:

```
05:36:52 [mysql] This may be due to a blocked port, missing dependencies,
05:36:52 [mysql] improper privileges, a crash, or a shutdown by another method.
05:36:52 [mysql] Press the Logs button to view error logs and check
05:36:52 [mysql] the Windows Event Viewer for more clues
05:36:52 [mysql] If you need more help, copy and post this
05:36:52 [mysql] entire log window on the forums
05:37:04 [mysql] Attempting to start MySQL app...
05:37:05 [mysql] Status change detected: running
```

**Bottom Screenshot: phpMyAdmin**

The phpMyAdmin interface shows the database structure for the 'sistemaventa' database. The left sidebar lists the databases: information\_schema, mysql, performance\_schema, phpmyadmin, and sistemaventa. The 'sistemaventa' database is selected, and the 'Estructura' (Structure) tab is active. The table list shows the following tables:

Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Residuo a depurar
clientes	Examinar, Estructura, Buscar, Insertar, Vaciar, Eliminar	1	InnoDB	utf8mb4_general_ci	16.0 KB	-
config	Examinar, Estructura, Buscar, Insertar, Vaciar, Eliminar	1	InnoDB	utf8mb4_general_ci	16.0 KB	-
detalle	Examinar, Estructura, Buscar, Insertar, Vaciar, Eliminar	20	InnoDB	utf8mb4_general_ci	16.0 KB	-
productos	Examinar, Estructura, Buscar, Insertar, Vaciar, Eliminar	3	InnoDB	utf8mb4_general_ci	16.0 KB	-
proveedor	Examinar, Estructura, Buscar, Insertar, Vaciar, Eliminar	1	InnoDB	utf8mb4_general_ci	16.0 KB	-
usuarios	Examinar, Estructura, Buscar, Insertar, Vaciar, Eliminar	3	InnoDB	utf8mb4_general_ci	16.0 KB	-
ventas	Examinar, Estructura, Buscar, Insertar, Vaciar, Eliminar	3	InnoDB	utf8mb4_general_ci	16.0 KB	-
<b>7 tablas</b>	<b>Número de filas</b>	<b>32</b>	<b>InnoDB</b>	<b>utf8mb4_general_ci</b>	<b>112.0 KB</b>	<b>0 B</b>

The interface also includes a search bar, a 'Crear nueva tabla' (Create new table) button, and a 'Seleccionar todo' (Select all) checkbox.