

# **PROYECTO POO “GESTION DE MASCOTAS”**

Jajaira Lizbeth Proaño Canchig

Instituto Superior Tecnológico

“CENESTUR”

**Programación orientada a objetos**

Ing. Yadira Franco

## **1. Introducción**

El sistema de Gestión de Mascotas es una aplicación CRUD desarrollada en Java Swing con conexión a MySQL.

El sistema está orientado a la gestión integral de dueños, mascotas, controles médicos y facturas.

Se implementa un control de roles:

- Administrador: acceso completo a todos los CRUDS.
- Cajero: acceso a mascotas, controles y facturas.

Este proyecto aplica conceptos de Programación Orientada a Objetos (POO), persistencia de datos, eventos de interfaz gráfica y JDBC.

## **2. Objetivos**

### **Objetivo General**

Desarrollar una aplicación CRUD en Java con interfaz gráfica y persistencia en MySQL para la gestión de una veterinaria.

### **Objetivos Específicos**

- Diseñar un modelo de base de datos relacional.
- Implementar conexión JDBC entre Java y MySQL.
- Desarrollar pantallas con Java Swing usando un patrón modular.
- Aplicar validaciones y control de acceso según rol.
- Documentar el desarrollo (manual técnico y capturas).

### 3. Arquitectura del Sistema

El proyecto sigue una arquitectura modular por paquetes:

```
src/
├── Conexion/           # Conexión a MySQL
│   └── ConexionBD.java
├── Ingreso/           # Login
│   └── Ingreso_usuario.java
├── Vista/             # Menús y CRUDs
│   ├── MenuAdmin.java
│   ├── MenuCajero.java
│   ├── CRUDDueno.java
│   ├── CRUDMascota.java
│   ├── CRUDControl.java
│   └── CRUDFactura.java
├── Main/              # Clase principal
│   └── Main.java
```

#### Características técnicas:

- **Separación por capas:** cada módulo tiene una responsabilidad clara.
- **Modularidad:** CRUDS independientes.
- **Reutilización de código:** conexión compartida mediante ConexionBD.

### 4. Herramienta Figt

El mockup de la interfaz de usuario se divide en cuatro secciones principales:

- Ingreso:** Contiene un formulario para el "Ingreso de cuenta" con campos para "Usuario" y "Contraseña", y un botón "Ingresar".
- Administración:** Muestra un "Menú Administración" con botones para "Gestionar dueño", "Gestionar Mascotas", "Controles médicos" y "Gestionar Reportes".
- Caja:** Presenta un "Menú de Caja" con botones para "Registro de mascota", "Registro de controles" y "Registro de factura".
- Mascota:** Incluye un formulario para el "Registro de mascota" con campos para "Dueño:", "Nombre de la mascota:", "Especie:" y "Raza:", un campo para "Foto" y un botón "Guardar".

## 5. Modelo de Datos

La base de datos gestion\_veterinaria contiene 5 tablas:

### 5.1 Estructura

#### Usuarios

```
-- Tabla de usuarios
CREATE TABLE IF NOT EXISTS usuarios (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nombre_usuario VARCHAR(50) NOT NULL,
  contraseña VARCHAR(100) NOT NULL,
  rol ENUM('admin', 'cajero') NOT NULL
);
INSERT INTO usuarios (id, nombre_usuario, contraseña, rol) VALUES
(01, 'Emily Lozada', '2018', 'admin'),
(02, 'Bryan Cisneros', '2021', 'cajero');
```

#### Dueños

```
-- Tabla de dueños
CREATE TABLE IF NOT EXISTS dueños (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(100),
  telefono VARCHAR(20),
  direccion VARCHAR(150)
);
```

#### Mascotas

```
-- Tabla de mascotas
CREATE TABLE IF NOT EXISTS mascotas (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(50),
  especie VARCHAR(50),
  raza VARCHAR(50),
  edad INT,
  id_dueno INT,
  FOREIGN KEY (id_dueno) REFERENCES dueños(id)
);
```

#### Controles médicos

```
-- Tabla de controles médicos
CREATE TABLE IF NOT EXISTS controles_medicos (
  id INT AUTO_INCREMENT PRIMARY KEY,
  fecha DATE,
  descripcion TEXT,
  veterinario VARCHAR(100),
  id_mascota INT,
  FOREIGN KEY (id_mascota) REFERENCES mascotas(id)
);
```

## Facturas

```
-- Tabla de facturas
CREATE TABLE IF NOT EXISTS facturas (
  id INT AUTO_INCREMENT PRIMARY KEY,
  fecha DATE,
  id_mascota INT,
  monto_total DECIMAL(10,2),
  FOREIGN KEY (id_mascota) REFERENCES mascotas(id)
);
```

## 5.2 Relaciones

- Dueño 1 —  $N$  Mascotas
- Mascota 1 —  $N$  Controles Médicos
- Mascota 1 —  $N$  Facturas

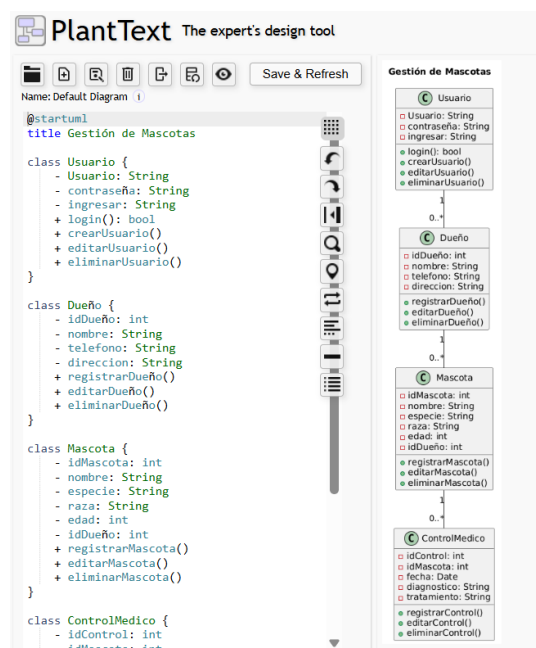
Las llaves foráneas (FOREIGN KEY) aseguran integridad referencial.

```
FOREIGN KEY (id_dueno) REFERENCES duenos(id)
```

```
FOREIGN KEY (id_mascota) REFERENCES mascotas(id)
```

```
FOREIGN KEY (id_mascota) REFERENCES mascotas(id)
```

## 6. Diagrama UML



## 7. Manual Técnico

### 7.1 ConexionBD

```
package Conexion;

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

public class ConexionBD {
    // URL de conexión a tu base de datos
    private static final String URL = "jdbc:mysql://localhost:3306/gestion_veterinaria";
    private static final String USER = "root";
    private static final String PASSWORD = "1234"; //mi contraseña

    // Método para conectar
    public static Connection conectar() {
        Connection conexion = null;
        try {
            conexion = DriverManager.getConnection(URL, USER, PASSWORD);
            System.out.println("Conexión exitosa a MySQL");
        } catch (SQLException e) {
            System.err.println("Error de conexión: " + e.getMessage());
        }
        return conexion;
    }
}
```

### 7.2 Ingreso\_usuario

- Atributos: txtUsuario, txtContraseña, btnIngresar.
- Métodos:
  - login() → Valida credenciales consultando usuarios.
  - Redirige a MenuAdmin o MenuCajero.

```
package Ingreso;

import Conexion.ConexionBD;
import java.sql.*;
import javax.swing.*;

public class Ingreso_usuario extends JFrame {
    private JTextField txtUsuario;
    private JPasswordField txtPassword;
    private JButton btnIngresar;

    public Ingreso_usuario() {
        setTitle("Ingreso de Cuenta");
        setSize(300, 200);
        setLayout(null);
        setLocationRelativeTo(null);

        JLabel lblUsuario = new JLabel("Usuario:");
        lblUsuario.setBounds(20, 20, 80, 25);
        add(lblUsuario);

        txtUsuario = new JTextField();
        txtUsuario.setBounds(100, 20, 150, 25);
        add(txtUsuario);

        JLabel lblPassword = new JLabel("Contraseña:");
        lblPassword.setBounds(20, 60, 80, 25);
        add(lblPassword);

        txtPassword = new JPasswordField();
        txtPassword.setBounds(100, 60, 150, 25);
        add(txtPassword);

        btnIngresar = new JButton("Ingresar");
        btnIngresar.setBounds(100, 100, 100, 25);
        btnIngresar.addActionListener(e -> login());
        add(btnIngresar);
    }
}
```

### 7.3 Menú administrador

- Botones para abrir CRUDDueno, CRUDMascota, CRUDControl.

```
package Vista;

import javax.swing.*;

public class MenuAdmin extends JFrame {

    public MenuAdmin() {
        setTitle("Menú Administración");
        setSize(300, 250);
        setLayout(null);
        setLocationRelativeTo(null);

        JButton btnDueno = new JButton("Gestionar Dueño");
        btnDueno.setBounds(50, 20, 200, 30);
        btnDueno.addActionListener(e -> new CRUDDueños());
        add(btnDueno);

        JButton btnMascota = new JButton("Gestionar Mascotas");
        btnMascota.setBounds(50, 60, 200, 30);
        btnMascota.addActionListener(e -> new CRUDMascotas());
        add(btnMascota);

        JButton btnControles = new JButton("Controles Médicos");
        btnControles.setBounds(50, 100, 200, 30);
        btnControles.addActionListener(e -> new CRUDControl());
        add(btnControles);

        JButton btnReportes = new JButton("Gestionar Reportes");
        btnReportes.setBounds(50, 140, 200, 30);
        add(btnReportes);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public static void main(String[] args) {
        new MenuAdmin();
    }
}
```

### 7.4 Menú Cajero

- Botones para abrir CRUDMascota, CRUDControl, CRUDFactura.

```
package Vista;

import javax.swing.*;

public class MenuCajero extends JFrame {

    public MenuCajero() {
        setTitle("Menú Cajero");
        setSize(300, 200);
        setLayout(null);
        setLocationRelativeTo(null);

        JButton btnMascota = new JButton("Registro Mascota");
        btnMascota.setBounds(50, 20, 200, 30);
        btnMascota.addActionListener(e -> new CRUDMascotas());
        add(btnMascota);

        JButton btnControl = new JButton("Registro Controles");
        btnControl.setBounds(50, 60, 200, 30);
        btnControl.addActionListener(e -> new CRUDControl());
        add(btnControl);

        JButton btnFactura = new JButton("Registro Factura");
        btnFactura.setBounds(50, 100, 200, 30);
        btnFactura.addActionListener(e -> new CRUDFactura());
        add(btnFactura);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public static void main(String[] args) {
        new MenuCajero();
    }
}
```

## 7.5 CRUDS

### CRUDDueño

- agregarDueño() → INSERT INTO duenos.
- actualizarDueño() → UPDATE duenos.
- eliminarDueño() → DELETE FROM duenos.
- cargarDueños() → SELECT \* FROM duenos.

```
package Vista;

import Conexion.ConexionBD;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.sql.*;

public class CRUDDueños extends JFrame {

    private JTable table;
    private DefaultTableModel model;
    private JTextField txtNombre, txtTelefono, txtDireccion;
    private JButton btnAgregar, btnActualizar, btnEliminar, btnCargar;

    public CRUDDueños() {
        setTitle("Gestión de Dueños");
        setSize(700, 400);
        setLocationRelativeTo(null);
        setLayout(new BorderLayout());

        // * Panel superior (formulario)
        JPanel panelForm = new JPanel();
        panelForm.setBorder(BorderFactory.createTitledBorder("Datos Dueño"));
        panelForm.setLayout(new GridBagLayout());
        GridBagConstraints gbc = new GridBagConstraints();
        gbc.insets = new Insets(5, 5, 5, 5); // Espaciado

        // Nombre
        gbc.gridx = 0; gbc.gridy = 0;
        panelForm.add(new JLabel("Nombre:"), gbc);

        gbc.gridx = 1;
        txtNombre = new JTextField(15);
        panelForm.add(txtNombre, gbc);

        // Telefono
        gbc.gridx = 0; gbc.gridy = 1;
        panelForm.add(new JLabel("Telefono:"), gbc);

        gbc.gridx = 1;
        txtTelefono = new JTextField(15);
        panelForm.add(txtTelefono, gbc);

        // Direccion
        gbc.gridx = 0; gbc.gridy = 2;
        panelForm.add(new JLabel("Direccion:"), gbc);

        gbc.gridx = 1;
        txtDireccion = new JTextField(15);
        panelForm.add(txtDireccion, gbc);

        // Botones
        gbc.gridx = 0; gbc.gridy = 3;
        panelForm.add(btnAgregar, gbc);

        gbc.gridx = 1;
        panelForm.add(btnActualizar, gbc);

        gbc.gridx = 2;
        panelForm.add(btnEliminar, gbc);

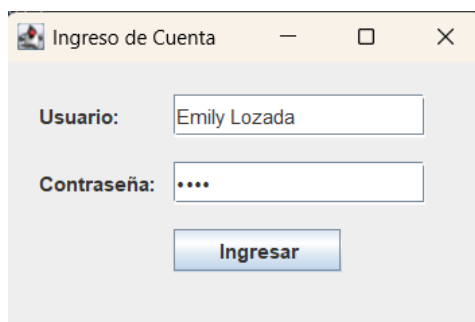
        gbc.gridx = 3;
        panelForm.add(btnCargar, gbc);

        // Tabla
        table = new JTable();
        model = new DefaultTableModel();
        table.setModel(model);
        table.setBounds(10, 410, 600, 150);
        add(table, BorderLayout.SOUTH);

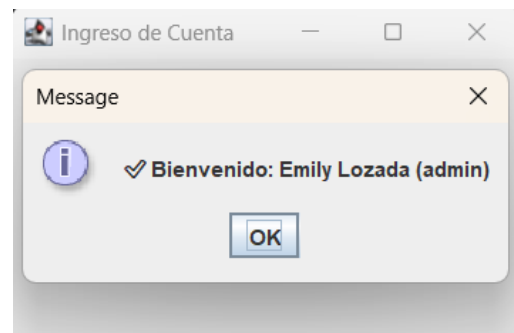
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

## 8. Documentación del Sistema

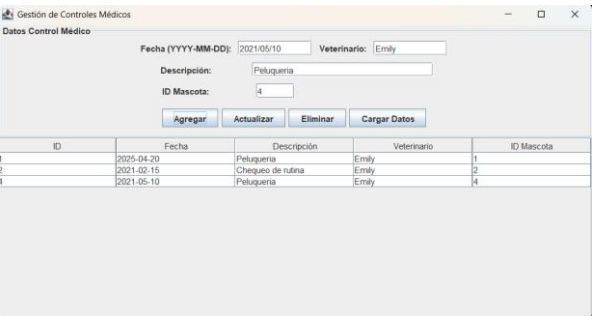
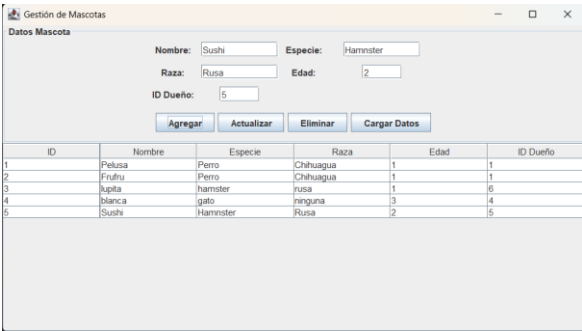
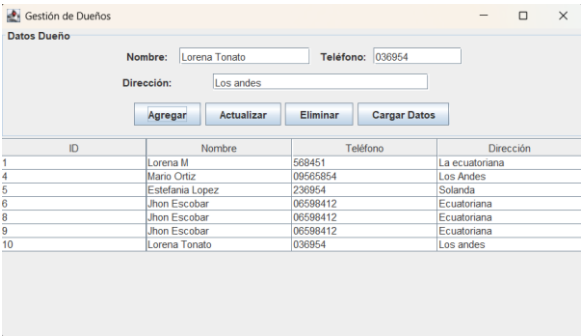
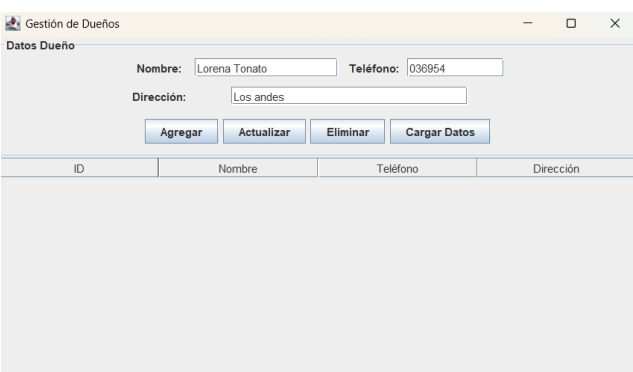
### Ingreso de cuenta administrativo



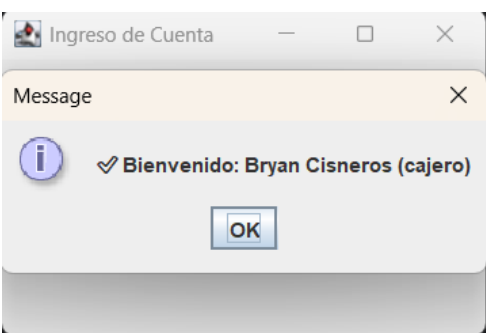
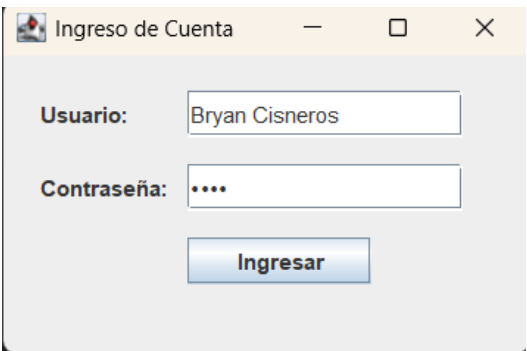
The screenshot shows a window titled "Ingreso de Cuenta". It contains two text input fields: "Usuario:" with the text "Emily Lozada" and "Contraseña:" with masked characters "....". Below the fields is a blue button labeled "Ingresar".



Menú administrativo

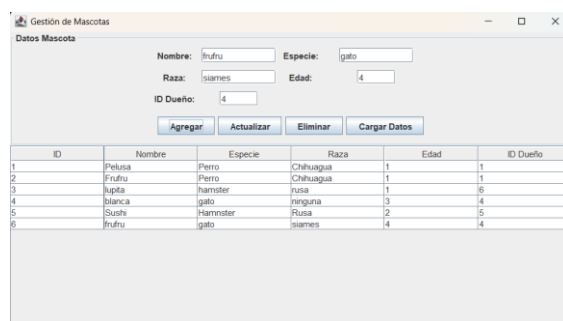
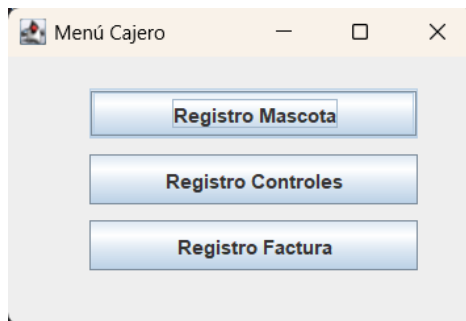


Ingreso de cuenta cajero





## Menú caja



The 'Gestión de Mascotas' window contains a form for pet data and a table of existing pets.

**Datos Mascota**

Nombre:  Especie:   
Raza:  Edad:   
ID Dueño:

ID	Nombre	Especie	Raza	Edad	ID Dueño
1	Pelusa	Perro	Chihuahua	1	1
2	Frufu	Perro	Chihuahua	1	1
3	luplo	hamster	rusa	1	8
4	blanca	gato	ninguna	3	4
5	Sushi	Hamster	Rusa	2	5
6	frufu	gato	siames	4	4

## 9. Conclusiones

El proyecto Gestión de Mascotas logró implementar con éxito un sistema CRUD completo con persistencia en MySQL, interfaz desarrollada en Java Swing y separación modular de componentes, cumpliendo con todos los requerimientos establecidos.