

Universidad Mariano Gálvez de Guatemala

Centro Universitario de Huehuetenango

Ingeniería en Sistemas de Información y Ciencias de la computación

Ing. Jhonny Morales Tello

Programación I

1do. Semestre

Ciclo 3

Sección: "A"

GRUPO 5

PROYECTO FINAL

Nombres

Carné

Alonzo Estuardo Alexander Morales García

0904-23-1037

Carlos Eduardo Gómez Reyes

0904-23-15175

Kimberly Analí Vásquez Rivas

0904-23-20310

Sindi Etelvina Castañeda Muñoz

0904-23-6875

Huehuetenango 31 de Mayo 2024

ÍNDICE

INTRODUCCIÓN	1
I. PROCESO DE ELABORACIÓN DE SOFTWARE	2
1. Descripción del sistema.....	2
2. Tecnologías utilizadas	2
2.1. Lenguaje de programación java.....	2
2.2. GitHub	2
2.3. NetBeans	3
2.4. MySQL	3
3. Planificación y diseño	3
3.1. Implementación	3
3.2. Pruebas	4
II. CÓDIGO FUENTE.....	4
1. Clase conexión	4
2. Clase consultas	5
3. Clase GetSetMascotas.....	9
4. Clase GetSetPersonas	11
5. Clase leer datos.....	12
6. VentanaMostrarAdoptante	13
7. VentanaMostrarMascotas	18

8.	VentanaPrincipalRefugio	23
9.	VentanaRegistroAdoptante	30
10.	VentanaRegistroMascotas	34
III.	MANUAL DE USUARIO	39
1.	Ventana Principal Refugio	39
2.	Registrar Adoptante.....	42
3.	Registrar Mascota	42
4.	Mostrar Adoptante.....	43
5.	Mostrar Mascota	43
IV.	CONCLUSIONES.....	44
V.	REPOSITORIO EN GITHUB	45

INTRODUCCIÓN

El desarrollo del sistema de control de mascotas se realizó con programación orientada a objetos (POO) siendo un área fundamental en la ingeniería de software para creación de distintos sistemas. La POO permite estructurar el código en objetos que representan entidades del mundo real, permite dividir el programa en tareas específicas, para su fácil modificación, reutilización y mantenimiento del software. En este proyecto, las clases y objetos serán utilizados para crear las entidades del refugio, como las mascotas y los adoptantes, encapsulando sus atributos.

El sistema se desarrolló en el lenguaje de programación Java y se está utilizando el manejo una base de datos para poder almacenar la información. Se implementarán las operaciones básicas de un CRUD (Crear, Leer, Actualizar y Eliminar), siendo esenciales para cualquier aplicación que requiera procesar datos. El sistema permitirá insertar datos de nuevas mascotas y adopciones, visualizar la lista de mascotas disponibles y el historial de adopciones, actualizar la información de las mascotas y el estado de las adopciones, y la eliminación de registros cuando sea necesario.

Para el desarrollo de este proyecto se utilizó GitHub para la colaboración del código, ya que es esencial para los futuros ingenieros de software tener la capacidad de poder apoyarse a través del trabajo en equipo. GitHub facilita el control de las distintas versiones de código y la integración continua de estos desde distintos lugares, permitiendo un desarrollo más organizado y eficiente.

I. PROCESO DE ELABORACIÓN DE SOFTWARE

1. Descripción del sistema

El sistema que desarrollamos tiene como objetivo principal gestionar la información de las mascotas que ingresan al refugio, así como también llevar un registro de las adopciones realizadas, incluyendo los datos tanto de las mascotas como de los adoptantes.

2. Tecnologías utilizadas

2.1. Lenguaje de programación java

Para la elaboración del sistema se utilizó el lenguaje de programación Java debido a sus características que lo convierten en una herramienta ideal para trabajar en este proyecto. Java facilita la creación del sistema y permite ajustarse a las necesidades específicas del refugio, ya que tiene la capacidad de manejar programación orientada a objetos. Puede ejecutarse en cualquier plataforma que tenga instalada la JVM, por lo que es de ayuda para su fácil portabilidad del sistema.

2.2. GitHub

La utilización de GitHub ha sido algo fundamental para su creación porque permite trabajar en equipo desde distintos lugares para mejorar el código, contribuir en las nuevas funcionalidades o para solucionar problemas del proyecto. Al crear un repositorio en GitHub, se puede almacenar todo el código del proyecto en un único lugar, lo que permitió a todos los miembros del equipo trabajar sobre la misma base de código. Esto garantiza que todos pueden trabajar al mismo tiempo sin conflictos, manteniendo la estabilidad del sistema y facilitando la colaboración efectiva.

2.3. NetBeans

Se eligió NetBeans como entorno de desarrollo por las ventajas que nos ofrece para el desarrollo del software. NetBeans proporciona herramientas que facilitan la creación, edición y mantenimiento del código, mejorando la productividad y adaptabilidad del proyecto de control de mascotas del refugio, una de sus características útil para este proyecto fue el autocompletado de código que ayudó a escribir más rápido el código y con menos errores del IDE.

2.4. MySQL

Para el almacenamiento y gestión de datos se utilizó MySQL como base de datos. Al utilizar MySQL permitió diseñar la estructura de base de datos eficiente para almacenar la información de las mascotas y adopciones. Se crearon tablas para presentar a las mascotas con sus atributos correspondientes como nombre, género, edad y raza, también se crearon tablas para las adopciones con los datos del adoptante y mascotas, y fecha de adopción. MySQL tiene gran compatibilidad con java lo que facilitó la integración del sistema con la base de datos. Esto permite realizar operaciones CRUD (crear, leer, actualizar, eliminar) de manera eficiente y segura.

3. Planificación y diseño

Se definió de manera más detallada la funcionalidad que debía ofrecer y las características que se debía tener y se crearon ideas para representar visualmente el programa, detallando los componentes principales y las interfaces.

3.1. Implementación

Se llevó a cabo dividiendo el proyecto en módulos pequeños y manejables para facilitar el desarrollo y colaboración entre los miembros. Utilizando el lenguaje Java por su facilidad de uso, **GitHub** para la colaboración en equipo, NetBeans como el entorno para una fácil escritura, edición y depuración del código, y MySQL para almacenar la información.

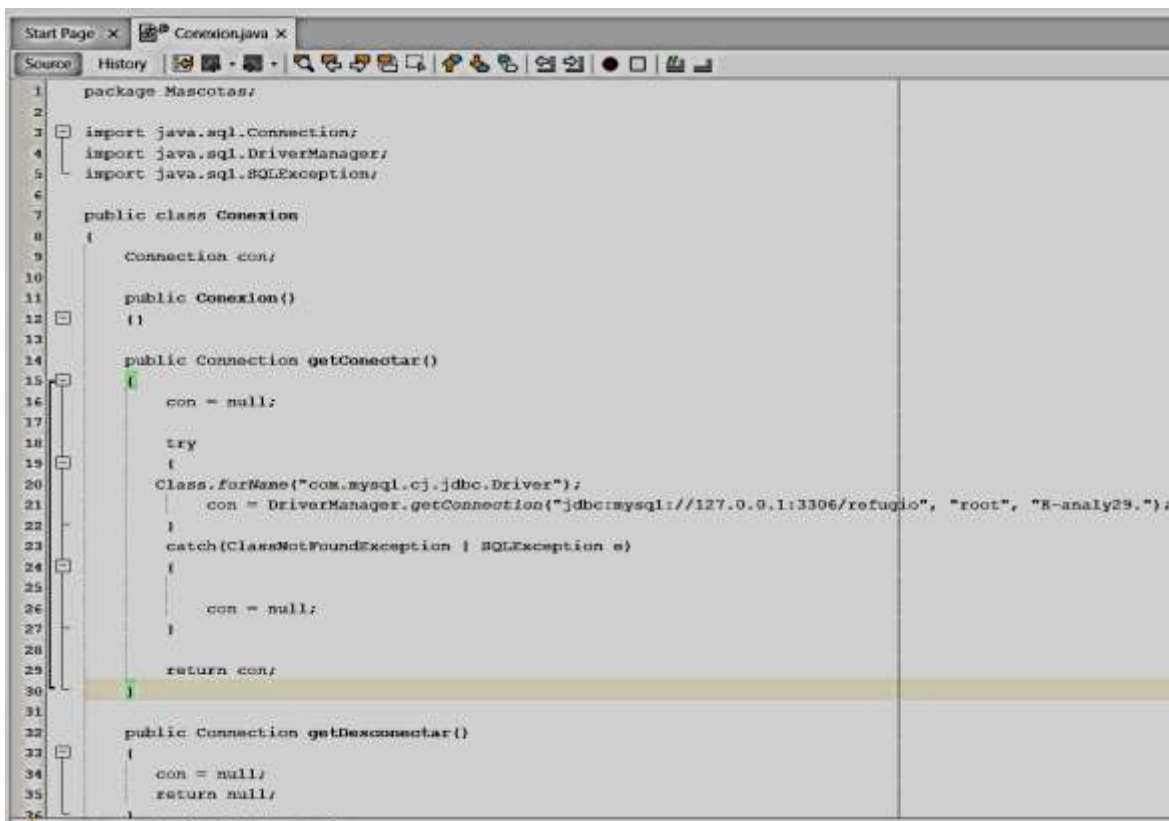
3.2. Pruebas

Se verificó el correcto funcionamiento del sistema al realizar pruebas para garantizar que el proyecto esté libre de errores y para verificar que cumpla con todos los requisitos definidos en la planificación.

II. CÓDIGO FUENTE

1. Clase conexión

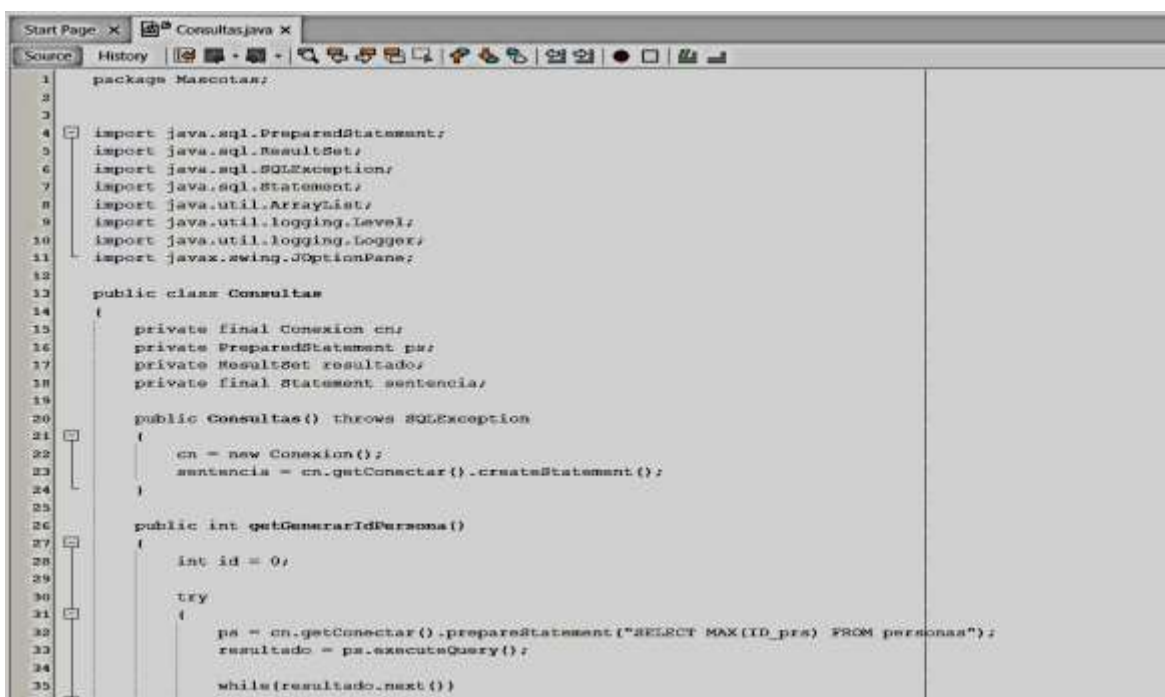
Se realizó una clase llamada “conexión”, el código se encuentra dentro del paquete “Mascotas” en donde se implementa la funcionalidad de conectarse a una base de datos MySQL. Esta clase proporciona dos métodos principales que serían getConectar() intenta establecer la conexión y la devuelve, y getDesconectar() que cierra la conexión a null.



```
1 package Mascotas;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 public class Conexion
8 {
9     Connection con;
10
11     public Conexion()
12     {}
13
14     public Connection getConectar()
15     {
16         con = null;
17
18         try
19         {
20             Class.forName("com.mysql.cj.jdbc.Driver");
21             con = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/refugio", "root", "R-analy29.");
22         }
23         catch(ClassNotFoundException | SQLException e)
24         {
25
26             con = null;
27         }
28
29         return con;
30     }
31
32     public Connection getDesconectar()
33     {
34         con = null;
35         return null;
36     }
37 }
```

2. Clase consultas

Se creó la clase Java llamada “Consultas” que se encarga de realizar operaciones relacionadas con la base de datos del sistema de control de un refugio de mascotas. Esta clase utiliza la otra clase llamada conexión para manejar la conexión con la base de datos MySQL, permitiendo tener acceso e interactuar con los datos del refugio de mascotas de manera segura.



```

1 package Mascotas;
2
3
4 import java.sql.PreparedStatement;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8 import java.util.ArrayList;
9 import java.util.logging.Level;
10 import java.util.logging.Logger;
11 import javax.swing.JOptionPane;
12
13 public class Consultas
14 {
15     private final Conexion cn;
16     private PreparedStatement ps;
17     private ResultSet resultado;
18     private final Statement sentencia;
19
20     public Consultas() throws SQLException
21     {
22         cn = new Conexion();
23         sentencia = cn.getConectar().createStatement();
24     }
25
26     public int getGenerarIdPersona()
27     {
28         int id = 0;
29
30         try
31         {
32             ps = cn.getConectar().prepareStatement("SELECT MAX(ID_pra) FROM personas");
33             resultado = ps.executeQuery();
34
35             while(resultado.next())

```



```

36             id = resultado.getInt(1) + 1;
37         }
38
39         ps.close();
40         resultado.close();
41         cn.getDesconectar();
42     }
43     catch (SQLException e)
44     {
45         JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
46     }
47     return id;
48 }
49
50 public boolean getInsertarPersonas(int ci, String nom, String gn, String dr, String tel) throws SQLException
51 {
52     try
53     {
54         ps = cn.getConectar().prepareStatement("INSERT INTO personas (ID_pra, nombre_pra, genero_pra, domicilio_pra, telefono_pra)"
55             + "VALUES (?, ?, ?, ?, ?)");
56
57         ps.setInt(1, ci);
58         ps.setString(2, nom);
59         ps.setString(3, gn);
60         ps.setString(4, dr);
61         ps.setString(5, tel);
62
63         ps.executeUpdate();
64         cn.getDesconectar();
65
66         return true;
67     }
68     catch (SQLException e)

```



```

        catch (SQLException e)
        {
            return false;
        }
    }

    public int getGeneraridMascota()
    {
        int id = 0;

        try
        {
            ps = cn.getConectar().prepareStatement("SELECT MAX(ID_mct) FROM mascotas");
            resultado = ps.executeQuery();

            while(resultado.next())
            {
                id = resultado.getInt(1) + 5;
            }

            ps.close();
            resultado.close();
            cn.getDesconectar();
        }
        catch (SQLException e)
        {
            JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
        }

        return id;
    }

    public boolean getLlenarMasootas(int idPer, int cl, String nom, int ed, String gn, String esp, String est)
    {
        try
        {

```

```

            resultado.next();

            nom = String.valueOf(resultado.getString("nombre_prs"));

            return nom;
        }
    }

    public ResultSet getLlenarTablas(String consulta)
    {
        try
        {
            resultado = sentencia.executeQuery(consulta);
        }
        catch (SQLException e)
        {
            JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
        }

        return resultado;
    }

    public String getConsultas(String consulta, String campo)
    {
        String cl = "";

        try
        {
            resultado = sentencia.executeQuery(consulta);

            resultado.next();

            cl = String.valueOf(resultado.getString(campo));
        }
        catch (SQLException ex)
        {

```

```

    try
    {
        ps = cn.getConnection().prepareStatement("SELECT MAX(ID_mct) FROM mascotas");
        resultado = ps.executeQuery();

        while(resultado.next())
        {
            id = resultado.getInt(1) + 5;
        }

        ps.close();
        resultado.close();
        cn.getConnection().close();
    }
    catch (SQLException e)
    {
        JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
    }

    return id;
}

public boolean getLlenarMascotas(int idPer, int cl, String nom, int ed, String gn, String esp, String est)
{
    try
    {
        ps = cn.getConnection().prepareStatement("INSERT INTO mascotas (ID_mct, nombre_mct, edad_mct, genero_mct, especie_mct, "
            + "estatus_mct, ID_prs) VALUES (?, ?, ?, ?, ?, ?, ?)");

        ps.setInt(1, cl);
        ps.setString(2, nom);
        ps.setInt(3, ed);
        ps.setString(4, gn);
        ps.setString(5, esp);
        ps.setString(6, est);
        ps.setInt(7, idPer);
    }

```

```

        ps.executeUpdate();
        cn.getConnection().close();

        return true;
    }
    catch (SQLException e)
    {
        return false;
    }
}

public ArrayList <String> getLlenarComboIdPersona() throws SQLException
{
    ArrayList <String> lista = new ArrayList <>();

    String consulta = "SELECT * FROM personas";

    resultado = sentencia.executeQuery(consulta);

    while(resultado.next())
    {
        lista.add(resultado.getString("ID_prs"));
    }

    return lista;
}

public String getNombrePersona(int id) throws SQLException
{
    String nom = "";

    String consulta = "SELECT * FROM personas WHERE ID_prs = '" + id + "'";

    resultado = sentencia.executeQuery(consulta);

```

```

        logger.getLogger(Consultas.class.getName()).log(Level.SEVERE, null, ex);
    }
    return cl;
}

public ArrayList<String> getMascotasPorMascotas(int id) throws SQLException
{
    ArrayList<String> lista = new ArrayList<>();

    String consulta = "SELECT * FROM mascotas where ID_pra = " + id;

    resultado = sentencia.executeQuery(consulta);

    while(resultado.next())
    {
        lista.add(resultado.getString("nombre_mct"));
    }

    return lista;
}

public boolean getModificarPersonas(int cl, String tel, String dir) throws SQLException
{
    try
    {
        String consulta = "UPDATE personas SET domicilio_pra = '" + dir + "', telefono_pra = '" + tel + "' WHERE ID_pra = " + cl;

        sentencia.executeUpdate(consulta);
        cn.getDesconectar();

        return true;
    }
    catch(SQLException e)
    {
        return false;
    }
}

```

```

public boolean getModificarMascotas(int cl, int ed, String est) throws SQLException
{
    try
    {
        String consulta = "UPDATE mascotas SET edad_mct = '" + ed + "', estatus_mct = '" + est + "' WHERE ID_mct = " + cl;

        sentencia.executeUpdate(consulta);
        cn.getDesconectar();

        return true;
    }
    catch(SQLException e)
    {
        return false;
    }
}

public boolean getEliminarMascotas(int cl)
{
    try
    {
        String consulta = "DELETE FROM mascotas WHERE ID_mct = " + cl;

        sentencia.executeUpdate(consulta);
        cn.getDesconectar();

        return true;
    }
    catch(SQLException e)
    {
        return false;
    }
}

public boolean getEliminarPersonas(int cl)
{
}

```

```

{
    try
    {
        String consulta1 = "DELETE FROM mascotas WHERE ID_pra = " + cl;
        String consulta2 = "DELETE FROM personas WHERE ID_pra = " + cl;

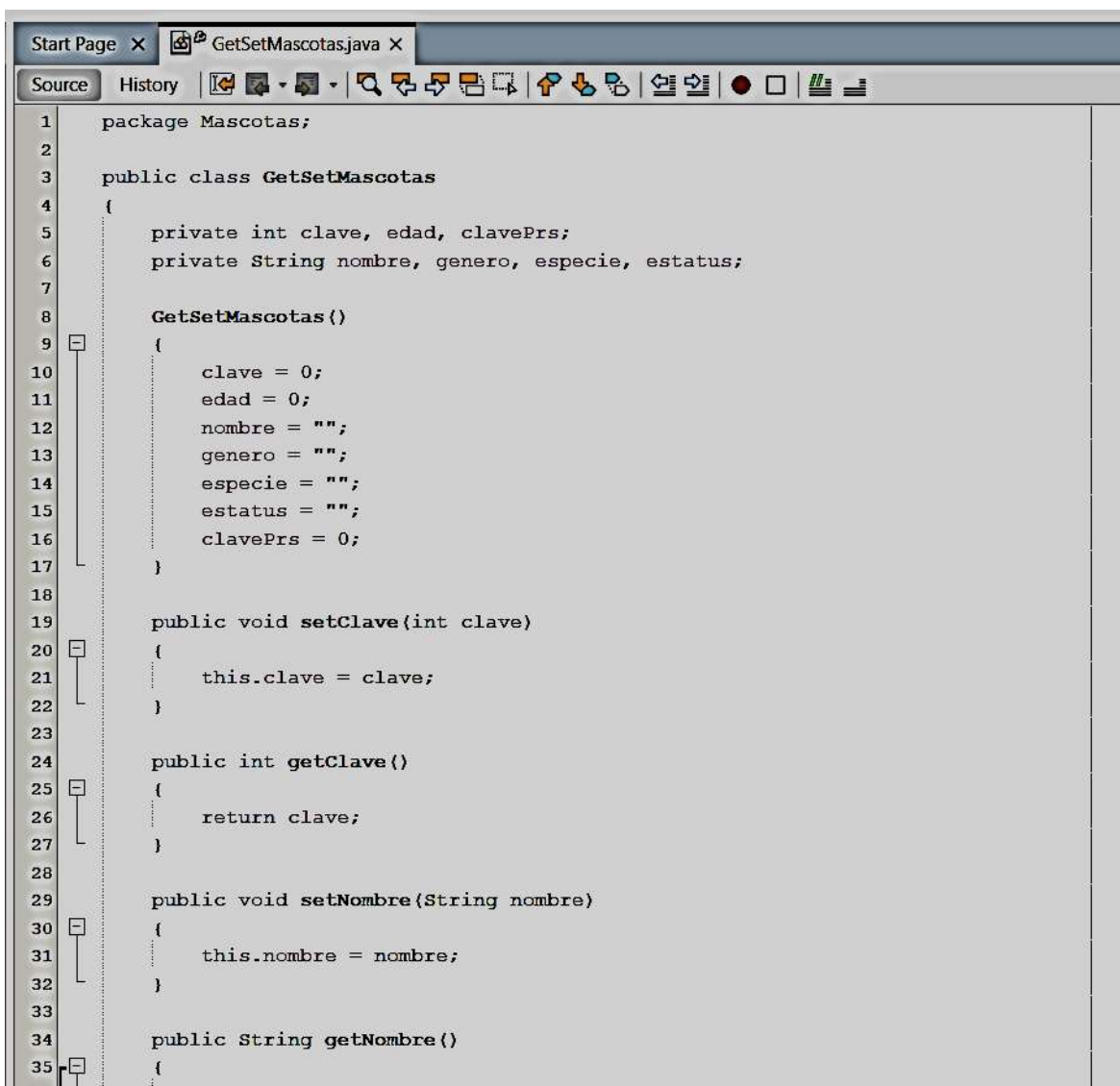
        sentencia.executeUpdate(consulta1);
        sentencia.executeUpdate(consulta2);
        cn.getDesconectar();

        return true;
    }
    catch(SQLException e)
    {
        return false;
    }
}
}

```

3. Clase GetSetMascotas

Se creó una nueva clase llamada “GetSetMascotas” que funciona como una entidad Mascota en donde se colocan sus respectivos atributos y métodos de acceso (getters y setters) para cada uno de sus atributos. Los métodos getter permiten obtener el valor de los atributos, mientras que los setter permiten establecer valor a los atributos. En esta clase se encapsulan los datos de una mascota, lo que proporciona una forma organizada de almacenar y acceder a la información con la mascota.



```
1 package Mascotas;
2
3 public class GetSetMascotas
4 {
5     private int clave, edad, clavePrs;
6     private String nombre, genero, especie, estatus;
7
8     GetSetMascotas()
9     {
10         clave = 0;
11         edad = 0;
12         nombre = "";
13         genero = "";
14         especie = "";
15         estatus = "";
16         clavePrs = 0;
17     }
18
19     public void setClave(int clave)
20     {
21         this.clave = clave;
22     }
23
24     public int getClave()
25     {
26         return clave;
27     }
28
29     public void setNombre(String nombre)
30     {
31         this.nombre = nombre;
32     }
33
34     public String getNombre()
35     {
```

```
public void setEdad(int edad)
{
    this.edad = edad;
}

public int getEdad()
{
    return edad;
}

public void setEspecie(String especie)
{
    this.especie = especie;
}

public String getEspecie()
{
    return especie;
}

public void setGenero(String genero)
{
    this.genero = genero;
}

public String getGenero()
{
    return genero;
}

public void setEstatus(String estatus)
{
    this.estatus = estatus;
}
```

```
public void setClavePrs(int clavePrs)
{
    this.clavePrs = clavePrs;
}

public int getClavePrs()
{
    return clavePrs;
}
```

4. Clase GetSetPersonas

Al igual que la clase “GetSetMascota” esta otra clase llamada “GetSetPersonas” tiene el código que se encarga de administrar los datos de las personas dentro del sistema, lo que proporciona una forma estructurada de almacenar, acceder y modificar la información sobre cada adoptante, también se utiliza el encapsulamiento.

```
package Mascotas;

public class GetSetPersonas
{
    private int clave;
    private String nombre, genero, direccion, telefono;

    public GetSetPersonas()
    {
        clave = 0;
        nombre = "";
        genero = "";
        direccion = "";
        telefono = "";
    }

    public void setClave(int clave)
    {
        this.clave = clave;
    }

    public int getClave()
    {
        return clave;
    }

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

    public String getNombre()
    {
        return nombre;
    }
}
```

```

    public void setGenero(String genero)
    {
        this.genero = genero;
    }

    public String getGenero()
    {
        return genero;
    }

    public void setDireccion(String direccion)
    {
        this.direccion = direccion;
    }

    public String getDireccion()
    {
        return direccion;
    }

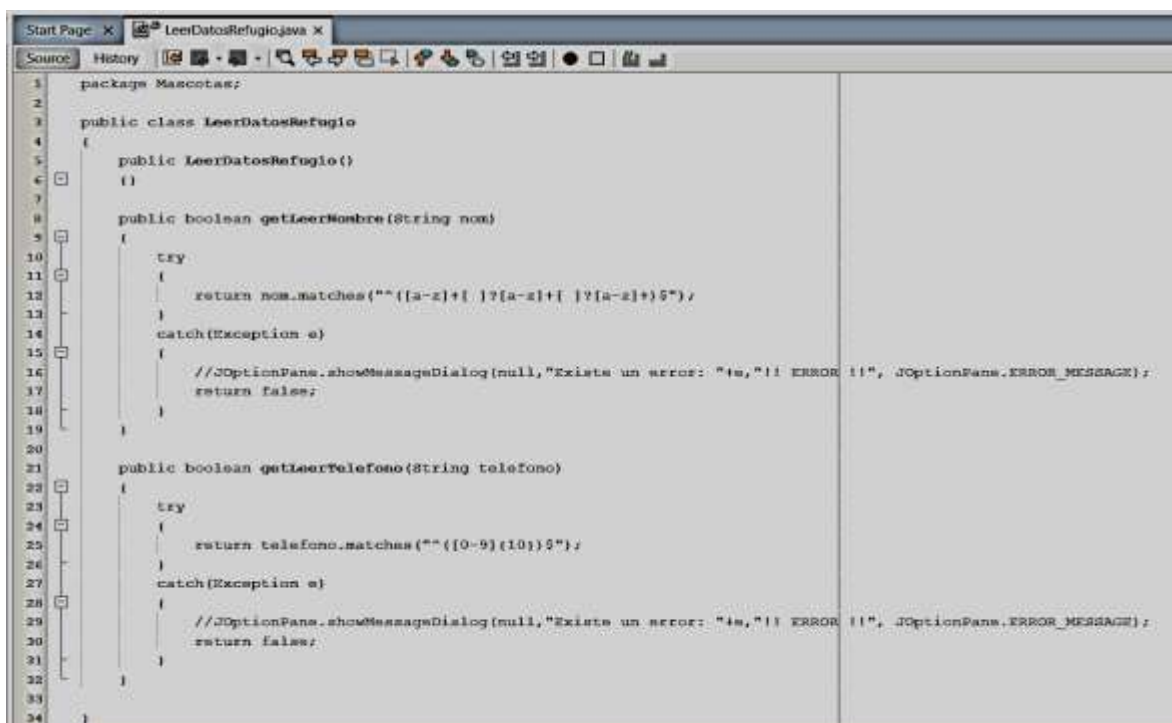
    public void setTelefono(String telefono)
    {
        this.telefono = telefono;
    }

    public String getTelefono()
    {
        return telefono;
    }
}

```

5. Clase leer datos

Esta clase es utilizada para validar datos de entrada antes de almacenarlos en la base de datos o procesarlos de alguna otra manera, de esta manera se verifica que los datos cumplan con el formato requerido.



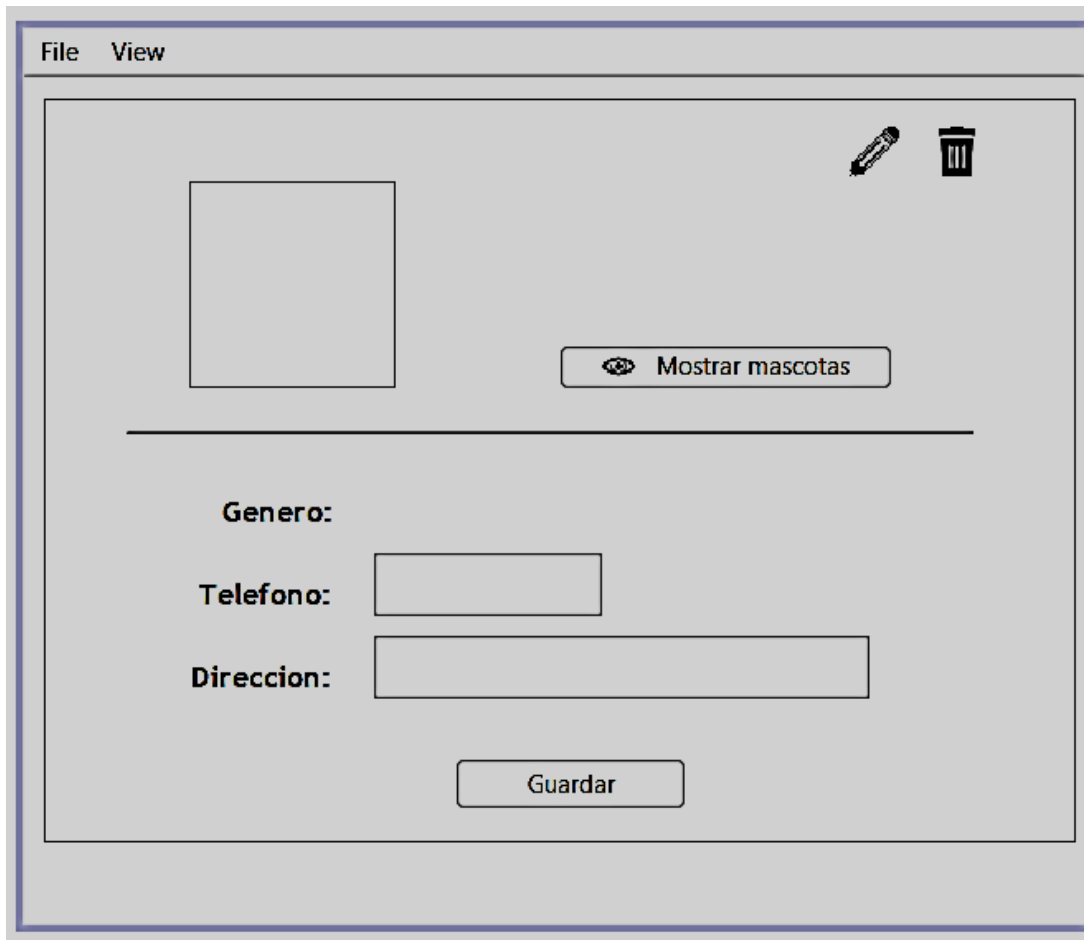
```

1 package Mascotas;
2
3 public class LeerDatosRefugio
4 {
5     public LeerDatosRefugio()
6     {
7     }
8
9     public boolean getLeerNombre(String nom)
10    {
11        try
12        {
13            return nom.matches("[a-z]{1} ?[a-z]{1} ?[a-z]{1}$");
14        }
15        catch (Exception e)
16        {
17            //JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", JOptionPane.ERROR_MESSAGE);
18            return false;
19        }
20    }
21
22    public boolean getLeerTelefono(String telefono)
23    {
24        try
25        {
26            return telefono.matches("[0-9]{10}$");
27        }
28        catch (Exception e)
29        {
30            //JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", JOptionPane.ERROR_MESSAGE);
31            return false;
32        }
33    }
34 }

```

6. VentanaMostrarAdoptante

Este código proporciona una ventana con la interfaz gráfica para mostrar, modificar y eliminar la información de los adoptantes del refugio. La información se obtiene y se actualiza en una base de datos.



```

1 package Mascotas;
2
3 import java.awt.Color;
4 import java.awt.Image;
5 import java.io.File;
6 import java.sql.SQLException;
7 import java.util.logging.Level;
8 import java.util.logging.Logger;
9 import javax.imageio.ImageIO;
10 import javax.swing.ImageIcon;
11 import javax.swing.JOptionPane;
12

```



```

12
13 public class VentanaMostrarAdoptante extends javax.swing.JFrame {
14
15     public VentanaMostrarAdoptante() throws SQLException
16     {
17         initComponents();
18
19         this.setLocationRelativeTo(null);
20
21         cn = new Conexion();
22         cl = new Consultas();
23
24         this.setConexion();
25
26         jTextField1.setVisible(false);
27         jTextField2.setVisible(false);
28
29         jButton1.setVisible(false);
30
31         //Icono de la Ventana
32         try{
33             Image img = ImageIO.read(new File("calendar.png"));
34             this.setIconImage(img);
35         }catch(Exception e)
36         {
37             JOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
38         }
39     }
40
41     public void setConexion()
42     {
43         if(cn.getConectar() != null)
44         {

```

```

44         {
45             jLabel1.setText("Conectado");
46             //jLabel9.setForeground(Color.GREEN);
47         }
48         else
49         {
50             jLabel1.setText("Desconectado");
51             jLabel1.setForeground(Color.red);
52         }
53     }
54
55     public void setPersona(int id)
56     {
57         String nombre = "", genero = "", domicilio = "", telefono = "";
58
59         String consulta = "SELECT * FROM personas WHERE ID_prs = '" + id + "'";
60
61         nombre = cl.getConsultas(consulta, "nombre_prs");
62         genero = cl.getConsultas(consulta, "genero_prs");
63         telefono = cl.getConsultas(consulta, "telefono_prs");
64         domicilio = cl.getConsultas(consulta, "domicilio_prs");
65
66         if(genero.equals("Masculino"))
67         {
68             jLabel2.setIcon(new ImageIcon("src/Imagenes/boy-broad-smile (2).png"));
69         }
70         else if(genero.equals("Femenino"))
71         {
72             jLabel2.setIcon(new ImageIcon("src/Imagenes/chica.png"));
73         }
74         else
75         {
76             jLabel2.setText("X");
77         }
78
79         jLabel4.setText(String.valueOf(id));
80         jLabel3.setText(nombre);
81         jLabel5.setText(genero);
82         jLabel6.setText(telefono);
83         jLabel7.setText(domicilio);
84     }

```

```

283 private void jLabel12MouseClicked(java.awt.event.MouseEvent evt) {
284     jLabel7.setVisible(false);
285
286     jTextField1.setVisible(true);
287     jTextField2.setVisible(true);
288
289     jButton1.setVisible(true);
290     jTextField1.setText(jLabel6.getText());
291     jTextField2.setText(jLabel7.getText());
292 }
293
294 private void jMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
295     System.exit(0);
296 }
297
298 private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {
299     try
300     {
301         VentanaPrincipalRefugio vpv = new VentanaPrincipalRefugio();
302
303         this.setVisible(false);
304         vpv.setVisible(true);
305     }
306     catch (SQLException e)
307     {
308         JOptionPane.showMessageDialog(null, "Existe un error: "+e, "!! ERROR !!", 2);
309     }
310 }
311
312 private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
313     try
314     {
315         VentanaMostrarMascotas vmm = new VentanaMostrarMascotas();
316
317         this.dispose();
318         vmm.setVisible(true);
319         vmm.setLenarComboMascotas(Integer.parseInt(jLabel4.getText()));
320     }
321     catch (SQLException e)
322     {
323         JOptionPane.showMessageDialog(null, "Existe un error: "+e, "!! ERROR !!", 2);
324     }
325 }
326
327 private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
328     try
329     {
330         VentanaMostrarMascotas vmm = new VentanaMostrarMascotas();
331
332         this.dispose();
333         vmm.setVisible(true);
334         vmm.setLenarComboMascotas(Integer.parseInt(jLabel4.getText()));
335     }
336     catch (SQLException e)
337     {
338         JOptionPane.showMessageDialog(null, "Existe un error: "+e, "!! ERROR !!", 2);
339     }
340 }
341
342 private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
343     int op = 0;
344
345     op = JOptionPane.showConfirmDialog(null, "¿Seguro desea modificar a " + jLabel3.getText() + "?",
346
347     if(op == JOptionPane.YES_OPTION)
348     {
349         if(jTextField1.getText().length() > 0 && jTextField2.getText().length() > 0)
350         {
351             try
352             {
353                 if(cl.getModificarPersonas(Integer.parseInt(jLabel4.getText()), jTextField1.getText())
354                 {
355                     JOptionPane.showMessageDialog(null, jLabel3.getText() + " ha sido modificado", "Co
356
357                     VentanaPrincipalRefugio vpv = new VentanaPrincipalRefugio();
358
359                     this.dispose();
360                     vpv.setVisible(true);
361                 }
362                 else
363                 {
364                     JOptionPane.showMessageDialog(null, "El contacto no se ha podido modificar", "!! ERR
365                 }
366             }
367             catch (SQLException e)
368             {
369                 JOptionPane.showMessageDialog(null, "Ha ocurrido un error: " +e, "!! ERROR !!", 2);
370             }
371         }
372     }
373 }

```

```

370         else
371             JOptionPane.showMessageDialog(null, "Los datos ingresados son incorrectos", "!! ERROR !!",
372             );
373         else
374         {
375             jTextField1.setVisible(false);
376             jTextField2.setVisible(false);
377
378             jLabel6.setVisible(true);
379             jLabel7.setVisible(true);
380
381             jButton1.setVisible(false);
382         }
383     }
384
385     private void jLabel11MouseClicked(java.awt.event.MouseEvent evt) {
386         int op = 0;
387
388         op = JOptionPane.showConfirmDialog(null, "¿Seguro desea eliminar a " + jLabel3.getText() + "?", "
389
390         if(op == JOptionPane.YES_OPTION)
391         {
392             if(c1.getEliminarPersonas(Integer.parseInt(jLabel4.getText())))
393             {
394                 JOptionPane.showMessageDialog(null, jLabel3.getText() + " ha sido eliminado", "Persona eli
395
396                 try
397                 {
398                     VentanaPrincipalRefugio vpv = new VentanaPrincipalRefugio();
399
400                     this.dispose();
401                     vpv.setVisible(true);
402                 }
403                 catch (SQLException e)
404                 {
405                     JOptionPane.showMessageDialog(null, "Ha ocurrido un error: " + e, "!! ERROR !!", 2);
406                 }
407             }
408             else
409                 JOptionPane.showMessageDialog(null, "La persona no se ha podido eliminar", "!! ERROR !!", 2
410             );
411         }
412     }

```

```

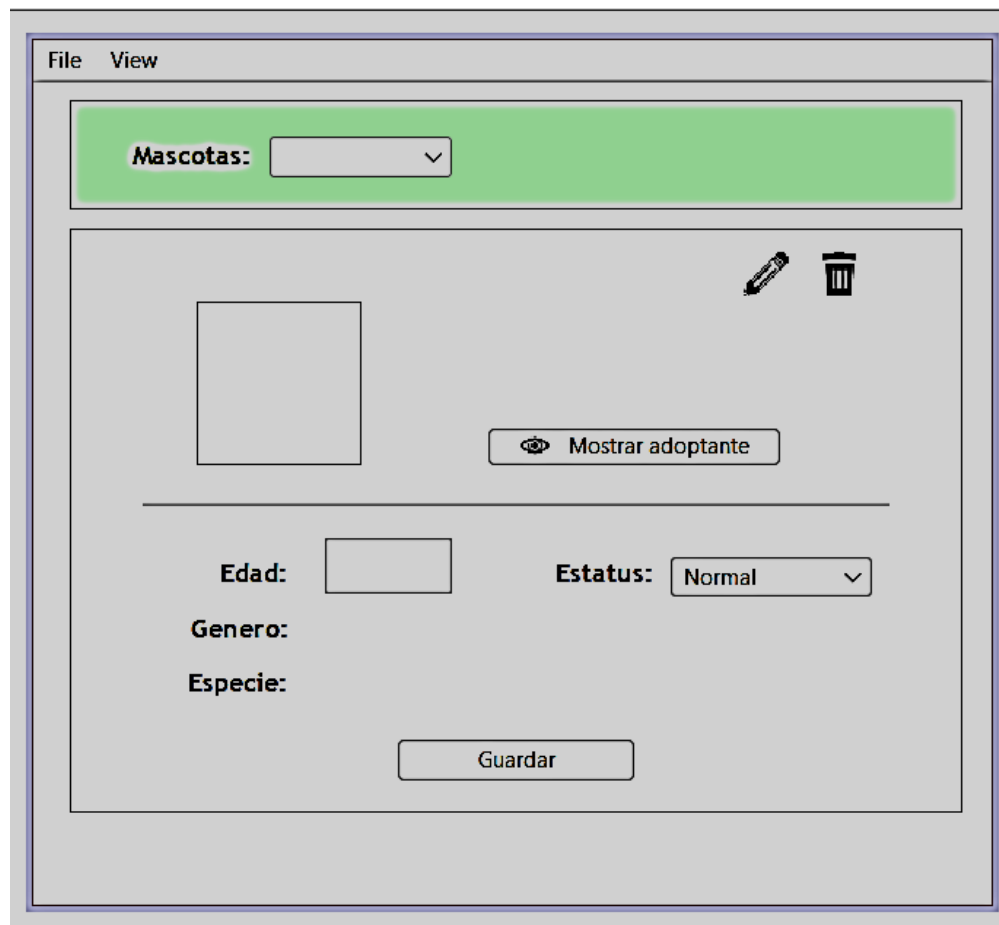
415     public static void main(String args[]) {
416
417         try {
418             for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
419                 if ("Nimbus".equals(info.getName())) {
420                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
421                     break;
422                 }
423             }
424         } catch (ClassNotFoundException ex) {
425             java.util.logging.Logger.getLogger(VentanaMostrarAdoptante.class.getName()).log(java.util.logging.Level.SEVERE,
426             );
427         } catch (InstantiationException ex) {
428             java.util.logging.Logger.getLogger(VentanaMostrarAdoptante.class.getName()).log(java.util.logging.Level.SEVERE,
429             );
430         } catch (IllegalAccessException ex) {
431             java.util.logging.Logger.getLogger(VentanaMostrarAdoptante.class.getName()).log(java.util.logging.Level.SEVERE,
432             );
433         } catch (javax.swing.UnsupportedLookAndFeelException ex) {
434             java.util.logging.Logger.getLogger(VentanaMostrarAdoptante.class.getName()).log(java.util.logging.Level.SEVERE,
435             );
436         }
437
438         java.awt.EventQueue.invokeLater(new Runnable() {
439             public void run() {
440                 try
441                 {
442                     new VentanaMostrarAdoptante().setVisible(true);
443                 }
444                 catch (SQLException e)
445                 {
446                     JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
447                 }
448             }
449         });
450     }

```

```
449
450 // Variables declaration - do not modify
451 private javax.swing.JButton jButton1;
452 private javax.swing.JButton jButton2;
453 private javax.swing.JLabel jLabel1;
454 private javax.swing.JLabel jLabel10;
455 private javax.swing.JLabel jLabel11;
456 private javax.swing.JLabel jLabel12;
457 private javax.swing.JLabel jLabel2;
458 private javax.swing.JLabel jLabel3;
459 private javax.swing.JLabel jLabel4;
460 private javax.swing.JLabel jLabel5;
461 private javax.swing.JLabel jLabel6;
462 private javax.swing.JLabel jLabel7;
463 private javax.swing.JLabel jLabel8;
464 private javax.swing.JLabel jLabel9;
465 private javax.swing.JMenu jMenu1;
466 private javax.swing.JMenu jMenu2;
467 private javax.swing.JMenuBar jMenuBar1;
468 private javax.swing.JMenuItem jMenuItem1;
469 private javax.swing.JMenuItem jMenuItem2;
470 private javax.swing.JMenuItem jMenuItem3;
471 private javax.swing.JPanel jPanel1;
472 private javax.swing.JPanel jPanel2;
473 private javax.swing.JPanel jPanel4;
474 private javax.swing.JSeparator jSeparator1;
475 private javax.swing.JTextField jTextField1;
476 private javax.swing.JTextField jTextField2;
477 // End of variables declaration
478 private Conexion cn;
479 private Consultas cl;
480 }
481
```


7. VentanaMostrarMascotas

Es una ventana gráfica que permite al usuario visualizar y gestionar la información de las mascotas del refugio de manera efectiva, siendo funcional debido a que interactúa con la base de datos para almacenar la información.



```

1 package Mascotas;
2
3 import java.awt.Color;
4 import java.awt.Image;
5 import java.io.File;
6 import java.sql.SQLException;
7 import java.util.ArrayList;
8 import java.util.logging.Level;
9 import java.util.logging.Logger;
10 import javax.imageio.ImageIO;
11 import javax.swing.ImageIcon;
12 import javax.swing.JOptionPane;
13

```

```

13
14 public class VentanaMostrarMascotas extends javax.swing.JFrame {
15
16     public VentanaMostrarMascotas() throws SQLException
17     {
18         initComponents();
19
20         this.setLocationRelativeTo(null);
21
22         cn = new Conexion();
23         cl = new Consultas();
24
25         this.setConexion();
26
27         jTextField1.setVisible(false);
28
29         jComboBox2.setVisible(false);
30
31         jButton2.setVisible(false);
32
33         jPanel4.setVisible(false);
34
35         //Icono de la Ventana
36         try{
37             Image img = ImageIO.read(new File("calendar.png"));
38             this.setIconImage(img);
39         }catch(Exception e)
40         {
41             JOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
42         }
43     }
44 ..

```

```

45     public void setConexion()
46     {
47         if(cn.getConectar() != null)
48         {
49             jLabel15.setText("Conectado");
50             //jLabel19.setForeground(Color.GREEN);
51         }
52         else
53         {
54             jLabel15.setText("Desconectado");
55             jLabel15.setForeground(Color.red);
56         }
57     }
58
59     public void setLlenarComboMascotas(int id)
60     {
61         ArrayList<String> lista = new ArrayList<String>();
62
63         try
64         {
65             jComboBox1.addItem("Selecciona");
66
67             lista = cl.getLlenarComboMascotas(id);
68
69             for(int i = 0; i < lista.size(); i++)
70                 jComboBox1.addItem(lista.get(i));
71
72         }
73         catch (SQLException e)
74         {
75             JOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
76         }
77     }
78
79     public void setMostrarMascotas(String nom)
80     {
81         String id = "", edad = "", genero = "", especie = "", estatus = "", propietario = "";
82
83         String consulta = "SELECT * FROM mascotas WHERE nombre_mct = '" + nom + "'";
84
85         id = cl.getConsultas(consulta, "ID_mct");
86         edad = cl.getConsultas(consulta, "edad_mct");
87         genero = cl.getConsultas(consulta, "genero_mct");
88         especie = cl.getConsultas(consulta, "especie_mct");

```

```

88     especie = cl.getConsultas(consulta, "especie_mct");
89     estatus = cl.getConsultas(consulta, "estatus_mct");
90     propietario = cl.getConsultas(consulta, "ID_prs");
91
92     jLabel5.setText(nom);
93     jLabel6.setText(id);
94     jLabel8.setText(edad);
95     jLabel10.setText(genero);
96     jLabel12.setText(especie);
97     jLabel14.setText(estatus);
98     jLabel16.setText(propietario);
99
100     if(especie.equals("Perro"))
101     {
102         jLabel11.setIcon(new ImageIcon("src/Imagenes/perro.png"));
103     }
104     else if(especie.equals("Gato"))
105     {
106         jLabel11.setIcon(new ImageIcon("src/Imagenes/gato.png"));
107     }
108     else if(especie.equals("Hamster"))
109     {
110         jLabel11.setIcon(new ImageIcon("src/Imagenes/hamster.png"));
111     }
112     else if(especie.equals("Pajaro"))
113     {
114         jLabel11.setIcon(new ImageIcon("src/Imagenes/pajaro.png"));
115     }
116     else
117     {
118         jLabel11.setText("X");
119     }
120
121 }

```

```

360 private void jLabel3MouseClicked(java.awt.event.MouseEvent evt) {
361     jLabel8.setVisible(false);
362     jLabel14.setVisible(false);
363
364     jTextField1.setVisible(true);
365     jComboBox2.setVisible(true);
366     jButton2.setVisible(true);
367
368     jTextField1.setText(jLabel8.getText());
369     jComboBox2.setSelectedItem(jLabel14.getText());
370 }
371
372 private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
373     System.exit(0);
374 }
375
376 private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
377     if(jComboBox1.getSelectedItem() == "Selecciona")
378     {
379         jPanel4.setVisible(false);
380         jMenuItem3.setEnabled(false);
381     }
382     else
383     {
384         jMenuItem3.setEnabled(true);
385         this.setMostrarMascotas((String) jComboBox1.getSelectedItem());
386         jPanel4.setVisible(true);
387     }
388 }
389
390 private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {
391     try
392     {
393         VentanaPrincipalRefugio vpv = new VentanaPrincipalRefugio();
394
395         this.setVisible(false);
396         vpv.setVisible(true);
397     }
398     catch (SQLException e)
399     {
400     }
401 }

```

```

402 JOOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
403 }
404 }
405
406 private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
407     try
408     {
409         VentanaMostrarAdoptante vmp = new VentanaMostrarAdoptante();
410
411         this.setVisible(false);
412         vmp.setVisible(true);
413         vmp.setPersona(Integer.parseInt(jLabel6.getText()));
414     }
415     catch (SQLException e)
416     {
417         JOOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
418     }
419 }
420
421 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
422     try
423     {
424         VentanaMostrarAdoptante vmp = new VentanaMostrarAdoptante();
425
426         this.setVisible(false);
427         vmp.setVisible(true);
428         vmp.setPersona(Integer.parseInt(jLabel6.getText()));
429     }
430     catch (SQLException e)
431     {
432         JOOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
433     }
434 }
435
436 private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
437     int op = 0;
438
439     op = JOOptionPane.showConfirmDialog(null, "¿Seguro desea modificar a " + jLabel5.getText() + "?", "Modificar", JOOption
440
441     if(op == JOOptionPane.YES_OPTION)
442     {
443         if(jTextField1.getText().length() > 0)
444

```

```

445         try
446         {
447             if(cl.getModificarMascotas(Integer.parseInt(jLabel6.getText()), Integer.parseInt(jTextField1.getText()),
448             {
449                 JOOptionPane.showMessageDialog(null, jLabel5.getText() + " ha sido modificado","Mascota modificada",
450
451                 VentanaPrincipalRefugio vpw = new VentanaPrincipalRefugio();
452
453                 this.dispose();
454                 vpw.setVisible(true);
455             }
456             else
457             {
458                 JOOptionPane.showMessageDialog(null,"La mascota no se ha podido modificar","!! ERROR !!", 2);
459             }
460             catch (SQLException e)
461             {
462                 JOOptionPane.showMessageDialog(null,"Ha ocurrido un error: "+e,"!! ERROR !!", 2);
463             }
464         }
465         else
466         {
467             JOOptionPane.showMessageDialog(null,"Los datos ingresados son incorrectos","!! ERROR !!", 2);
468         }
469     }
470     else
471     {
472         jTextField1.setVisible(false);
473         jComboBox2.setVisible(false);
474
475         jLabel8.setVisible(true);
476         jLabel14.setVisible(true);
477
478         jButton2.setVisible(false);
479     }
480 }
481
482 private void jButton4MouseClicked(java.awt.event.MouseEvent evt) {
483     int op = 0;
484
485     op = JOOptionPane.showConfirmDialog(null, "¿Seguro desea eliminar a " + jLabel5.getText() + "?", "Eliminar", JOOption
486
487     if(op == JOOptionPane.YES_OPTION)
488     {
489         if(cl.getEliminarMascotas(Integer.parseInt(jLabel6.getText())))
490         {
491             JOOptionPane.showMessageDialog(null, jLabel5.getText() + " ha sido eliminado","Mascota eliminada", 1);

```



```

488 JOptionPane.showMessageDialog(null, jLabel5.getText() + " ha sido eliminado", "Mascota eliminada", 1);
489
490 try
491 {
492     VentanaPrincipalRefugio vpw = new VentanaPrincipalRefugio();
493     this.dispose();
494     vpw.setVisible(true);
495 }
496 catch (SQLException e)
497 {
498     JOptionPane.showMessageDialog(null, "Ha ocurrido un error: " + e, "!! ERROR !!", 2);
499 }
500
501 else
502 {
503     JOptionPane.showMessageDialog(null, "La mascota no se ha podido eliminar", "!! ERROR !!", 2);
504 }
505
506
507 /**
508  * @param args the command line arguments
509  */
510 public static void main(String args[]) {
511
512     try
513     {
514         new VentanaMostrarMascotas().setVisible(true);
515     }
516     catch (SQLException e)
517     {
518         JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
519     }
520 }
521

```

```

523 // Variables declaration - do not modify
524 private javax.swing.JButton jButton1;
525 private javax.swing.JButton jButton2;
526 private javax.swing.JComboBox jComboBox1;
527 private javax.swing.JComboBox jComboBox2;
528 private javax.swing.JLabel jLabel1;
529 private javax.swing.JLabel jLabel10;
530 private javax.swing.JLabel jLabel11;
531 private javax.swing.JLabel jLabel12;
532 private javax.swing.JLabel jLabel13;
533 private javax.swing.JLabel jLabel14;
534 private javax.swing.JLabel jLabel15;
535 private javax.swing.JLabel jLabel16;
536 private javax.swing.JLabel jLabel2;
537 private javax.swing.JLabel jLabel3;
538 private javax.swing.JLabel jLabel4;
539 private javax.swing.JLabel jLabel5;
540 private javax.swing.JLabel jLabel6;
541 private javax.swing.JLabel jLabel7;
542 private javax.swing.JLabel jLabel8;
543 private javax.swing.JLabel jLabel9;
544 private javax.swing.JMenu jMenu1;
545 private javax.swing.JMenu jMenu2;
546 private javax.swing.JMenuBar jMenuBar1;
547 private javax.swing.JMenuItem jMenuItem1;
548 private javax.swing.JMenuItem jMenuItem2;
549 private javax.swing.JMenuItem jMenuItem3;
550 private javax.swing.JPanel jPanel1;
551 private javax.swing.JPanel jPanel2;
552 private javax.swing.JPanel jPanel3;
553 private javax.swing.JPanel jPanel4;
554 private javax.swing.JSeparator jSeparator1;
555 private javax.swing.JTextField jTextField1;
556 // End of variables declaration
557 private Conexion cn;
558 private Consultas cl;
559 }

```

8. VentanaPrincipalRefugio

Esta ventana permite la gestión de los datos del refugio de mascotas, en el que se encuentra la interfaz completa para manejar los distintos tipos de datos como agregar adopciones y mascotas, permite interactuar con las bases de datos para mostrar los datos de los adoptantes y mascotas.



```

1  package Mascotas;
2
3  import java.awt.Color;
4  import java.awt.Font;
5  import java.awt.Image;
6  import java.awt.event.KeyAdapter;
7  import java.awt.event.KeyEvent;
8  import java.io.File;
9  import java.sql.ResultSet;
10 import java.sql.SQLException;
11 import java.util.logging.Level;
12 import java.util.logging.Logger;
13 import javax.imageio.ImageIO;
14 import javax.swing.JOptionPane;
15 import javax.swing.RowFilter;
16 import javax.swing.table.DefaultTableCellRenderer;
17 import javax.swing.table.DefaultTableModel;
18 import javax.swing.table.JTableHeader;
19 import javax.swing.table.TableRowSorter;
20

```

```

20
21 public class VentanaPrincipalRefugio extends javax.swing.JFrame {
22
23     public VentanaPrincipalRefugio() throws SQLException
24     {
25         initComponents();
26
27         cn = new Conexion();
28         cl = new Consultas();
29
30         this.setLocationRelativeTo(null);
31
32         this.setConexion();
33
34         //Tabla de personas
35         th1 = jTable1.getTableHeader();
36
37         dtm1 = new DefaultTableModel(filas1, columnas1)
38         {
39             public boolean isCellEditable(int fil, int col)
40             {
41                 return false;
42             }
43         };

```

```

45         DefaultTableCellRenderer renderer1 = (DefaultTableCellRenderer)
46         jTable1.getTableHeader().getDefaultRenderer();
47
48         renderer1.setHorizontalAlignment(0);
49
50         th1.setReorderingAllowed(false);
51
52         th1.setFont(new Font("Calibri", Font.PLAIN, 13));
53
54         th1.setForeground(Color.BLACK);
55
56         th1.setResizingAllowed(false);
57
58         dtm1.fireTableDataChanged();
59
60         jTable1.setModel(dtm1);
61
62         this.setLlenarTablaPersonas("SELECT ID_prs, nombre_prs FROM personas");
63
64         //Tabla de mascotas
65         th2 = jTable2.getTableHeader();
66
67         dtm2 = new DefaultTableModel(filas2, columnas2)
68         {
69             public boolean isCellEditable(int fil, int col)
70             {
71                 return false;
72             }
73         };
74
75         DefaultTableCellRenderer renderer2 = (DefaultTableCellRenderer)
76         jTable2.getTableHeader().getDefaultRenderer();
77
78         renderer2.setHorizontalAlignment(0);
79
80         th2.setReorderingAllowed(false);
81
82         th2.setFont(new Font("Calibri", Font.PLAIN, 13));
83
84         th2.setForeground(Color.BLACK);
85
86         th2.setResizingAllowed(false);
87
88         dtm2.fireTableDataChanged();

```

```

89     jTable2.setModel(dtal2);
90
91     this.setLlenarTablaMascotas("SELECT ID_mct, nombre_mct, ID_pra FROM mascotas");
92
93     //Icono de la Ventana
94     try{
95         Image img = ImageIO.read(new File("calendar.png"));
96         this.setIconImage(img);
97     }catch(Exception e)
98     {
99         JOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
100     }
101
102
103
104     public void setConexion()
105     {
106         if(cn.getConectar() != null)
107         {
108             jLabel11.setText("Conectado");
109             //jLabel19.setForeground(Color.GREEN);
110         }
111         else
112         {
113             jLabel11.setText("Desconectado");
114             jLabel11.setForeground(Color.red);
115         }
116     }
117
118     public void setLlenarTablaPersonas(String consulta)
119     {
120         try
121         {
122             resultado1 = cl.getLlenarTablas(consulta);
123
124             while(resultado1.next())
125             {
126                 dtal.addRow(new Object[] {resultado1.getInt("ID_pra"), resultado1.getString("nombre_pra")});
127             }
128         }
129         catch (SQLException e)
130         {
131             JOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
132         }
133     }
134
135     public void setLlenarTablaMascotas(String consulta)
136     {
137         try
138         {
139             resultado2 = cl.getLlenarTablas(consulta);
140
141             while(resultado2.next())
142             {
143                 dtal.addRow(new Object[] {resultado2.getInt("ID_mct"), resultado2.getString("nombre_mct"), resultado2.getInt("ID_pra")});
144             }
145         }
146         catch (SQLException e)
147         {
148             JOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
149         }
150     }
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178

```

```

Generated Code
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2
```

```

479 {
480     VentanaRegistroMascotas vam = new VentanaRegistroMascotas();
481
482     this.dispose();
483     vam.setVisible(true);
484 } catch (SQLException e)
485 {
486     JOptionPane.showMessageDialog(null, "Error: " + e, "!! ERROR !!", 2);
487 }
488 }
489
490 private void jMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
491     System.exit(0);
492 }
493
494 private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
495     try
496     {
497         VentanaRegistroMascotas vam = new VentanaRegistroMascotas();
498
499         this.dispose();
500         vam.setVisible(true);
501     }
502     catch (SQLException e)
503     {
504         JOptionPane.showMessageDialog(null, "Error: " + e, "!! ERROR !!", 2);
505     }
506 }
507
508 private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
509     try
510     {
511         VentanaRegistroAdoptante vap = new VentanaRegistroAdoptante();
512
513         vap.setVisible(true);
514         this.dispose();
515     }
516     catch (SQLException e)
517     {
518         JOptionPane.showMessageDialog(null, "Error: " + e, "!! ERROR !!", 2);
519     }
520 }

```

```

521 private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
522     int fila = jTable1.getSelectedRow();
523     String campoSel = "";
524
525     try
526     {
527         VentanaMostrarAdoptante vmp = new VentanaMostrarAdoptante();
528
529         if(fila == -1)
530         {
531             //
532         }
533         else
534         {
535             campoSel = jTable1.getValueAt(fila, 0).toString();
536             this.setVisible(false);
537             vmp.setVisible(true);
538             vmp.setPersona(Integer.parseInt(campoSel));
539         }
540     }
541     catch (SQLException e)
542     {
543         JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
544     }
545 }
546
547 private void jTextField1KeyTyped(java.awt.event.KeyEvent evt) {
548     char car = evt.getKeyChar();
549
550     if((car<'0' || car>'9'))
551     {
552         evt.consume();
553     }
554
555     TableRowSorter trs = new TableRowSorter(dtml);
556
557     jTextField1.addKeyListener(new KeyAdapter() {
558         @Override
559         public void keyReleased(KeyEvent e) {
560             trs.setRowFilter(RowFilter.regexFilter(jTextField1.getText(), 0));
561         }
562     });
563
564     jTable1.setRowSorter(trs);

```



```

568 private void jTextField2KeyTyped(java.awt.event.KeyEvent evt) {
569     char car = evt.getKeyChar();
570
571     if((car<'0' || car>'9'))
572         evt.consume();
573
574     TableRowSorter trs = new TableRowSorter(dtm2);
575
576     jTextField2.addKeyListener(new KeyAdapter() {
577         @Override
578         public void keyReleased(KeyEvent e) {
579             trs.setRowFilter(RowFilter.regexFilter(jTextField2.getText(), 0));
580         }
581     });
582
583     jTable2.setRowSorter(trs);
584
585 }
586
587 private void jTable2MouseClicked(java.awt.event.MouseEvent evt) {
588     int fila = jTable2.getSelectedRow();
589     String campoSel = "";
590
591     try
592     {
593         VentanaMostrarMascotas vmm = new VentanaMostrarMascotas();
594
595         if(fila == -1)
596         {
597
598         }
599         else
600         {
601             campoSel = jTable2.getValueAt(fila, 2).toString();
602
603             this.dispose();
604             vmm.setVisible(true);
605             vmm.setLlenarComboMascotas(Integer.parseInt(campoSel));
606         }
607     }
608 }

```

```

609 catch (SQLException e)
610 {
611     JOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
612 }
613
614 private void jComboBox2ActionPerformed(java.awt.event.ActionEvent evt) {
615     int a = jTable1.getRowCount()-1;
616
617     if(jComboBox2.getSelectedItem().equals("Femenino"))
618     {
619         for (int i = a; i >= 0; i--)
620         {
621             dtal.removeRow(dtal.getRowCount()-1);
622         }
623
624         this.setLlenarTablaPersonas("SELECT ID_pra, nombre_pra FROM personas WHERE genero_pra = 'Femenino'");
625     }
626     else if(jComboBox2.getSelectedItem().equals("Masculino"))
627     {
628         for (int i = a; i >= 0; i--)
629         {
630             dtal.removeRow(dtal.getRowCount()-1);
631         }
632
633         this.setLlenarTablaPersonas("SELECT ID_pra, nombre_pra FROM personas WHERE genero_pra = 'Masculino'");
634     }
635 }
636
637 private void jLabel1MouseClicked(java.awt.event.MouseEvent evt) {
638     int a = jTable1.getRowCount()-1;
639
640     jComboBox2.setSelectedIndex(0);
641
642     for (int i = a; i >= 0; i--)
643     {
644         dtal.removeRow(dtal.getRowCount()-1);
645     }
646
647     this.setLlenarTablaPersonas("SELECT ID_pra, nombre_pra FROM personas");
648 }
649
650

```

```

652 private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
653     int a = jTable1.getRowCount()-1;
654     JComboBox2.setSelectedIndex(0);
655     for (int i = a; i >= 0; i--)
656     {
657         dtm1.removeRow(dtm1.getRowCount()-1);
658     }
659     this.setLlenarTablaPersonas("SELECT ID_pra, nombre_pra FROM personas");
660     int b = jTable2.getRowCount()-1;
661     JComboBox1.setSelectedIndex(0);
662     for (int i = b; i >= 0; i--)
663     {
664         dtm2.removeRow(dtm2.getRowCount()-1);
665     }
666     this.setLlenarTablaMascotas("SELECT ID_mct, nombre_mct, ID_pra FROM mascotas");
667 }
668
669 private void JComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {
670     int a = jTable2.getRowCount()-1;
671     if (jComboBox1.getSelectedItem().equals("Normal"))
672     {
673         for (int i = a; i >= 0; i--)
674         {
675             dtm2.removeRow(dtm2.getRowCount()-1);
676         }
677         this.setLlenarTablaMascotas("SELECT ID_mct, nombre_mct, ID_pra FROM mascotas WHERE estatus_mct = 'Normal'");
678     }
679     else if (jComboBox1.getSelectedItem().equals("Enfermo"))
680     {
681         for (int i = a; i >= 0; i--)
682         {
683             dtm2.removeRow(dtm2.getRowCount()-1);
684         }
685     }
686 }

```

```

694     this.setLlenarTablaMascotas("SELECT ID_mct, nombre_mct, ID_pra FROM mascotas WHERE estatus_mct = 'Enfermo'");
695 }
696 else if (jComboBox1.getSelectedItem().equals("Fallecido"))
697 {
698     for (int i = a; i >= 0; i--)
699     {
700         dtm2.removeRow(dtm2.getRowCount()-1);
701     }
702     this.setLlenarTablaMascotas("SELECT ID_mct, nombre_mct, ID_pra FROM mascotas WHERE estatus_mct = 'Fallecido'");
703 }
704
705 }
706
707 private void jLabel14MouseClicked(java.awt.event.MouseEvent evt) {
708     int a = jTable2.getRowCount()-1;
709     JComboBox1.setSelectedIndex(0);
710     for (int i = a; i >= 0; i--)
711     {
712         dtm2.removeRow(dtm2.getRowCount()-1);
713     }
714     this.setLlenarTablaMascotas("SELECT ID_mct, nombre_mct, ID_pra FROM mascotas");
715 }
716
717 private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {
718 }
719
720 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
721 }
722
723 }

```

```

731  /**
732   * @param args the command line arguments
733   */
734  public static void main(String args[]) {
735
736      try {
737          for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
738              if ("Winbus".equals(info.getName())) {
739                  javax.swing.UIManager.setLookAndFeel(info.getClassName());
740                  break;
741              }
742          }
743
744          catch (ClassNotFoundException ex) {
745              java.util.logging.Logger.getLogger(VentanaPrincipalRefugio.class.getName()).log(java.util.logging.Level.SEVERE,
746          ) catch (InstantiationException ex) {
747              java.util.logging.Logger.getLogger(VentanaPrincipalRefugio.class.getName()).log(java.util.logging.Level.SEVERE,
748          ) catch (IllegalAccessException ex) {
749              java.util.logging.Logger.getLogger(VentanaPrincipalRefugio.class.getName()).log(java.util.logging.Level.SEVERE,
750          ) catch (javax.swing.UnsupportedLookAndFeelException ex) {
751              java.util.logging.Logger.getLogger(VentanaPrincipalRefugio.class.getName()).log(java.util.logging.Level.SEVERE,
752          }
753
754      java.awt.EventQueue.invokeLater(new Runnable() {
755          public void run() {
756              try {
757                  new VentanaPrincipalRefugio().setVisible(true);
758              } catch (SQLException e) {
759                  JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
760              }
761          }
762      });
763  }
764
765

```

```

771  private javax.swing.JComboBox jComboBox2;
772  private javax.swing.JLabel jLabel1;
773  private javax.swing.JLabel jLabel11;
774  private javax.swing.JLabel jLabel12;
775  private javax.swing.JLabel jLabel13;
776  private javax.swing.JLabel jLabel14;
777  private javax.swing.JLabel jLabel2;
778  private javax.swing.JLabel jLabel3;
779  private javax.swing.JLabel jLabel4;
780  private javax.swing.JLabel jLabel5;
781  private javax.swing.JLabel jLabel6;
782  private javax.swing.JLabel jLabel7;
783  private javax.swing.JMenu jMenu1;
784  private javax.swing.JMenu jMenu2;
785  private javax.swing.JMenuBar jMenuBar1;
786  private javax.swing.JMenuItem jMenuItem1;
787  private javax.swing.JMenuItem jMenuItem2;
788  private javax.swing.JMenuItem jMenuItem3;
789  private javax.swing.JMenuItem jMenuItem4;
790  private javax.swing.JPanel jPanel10;
791  private javax.swing.JPanel jPanel11;
792  private javax.swing.JPanel jPanel19;
793  private javax.swing.JScrollPane jScrollPane1;
794  private javax.swing.JScrollPane jScrollPane2;
795  private javax.swing.JTabbedPane jTabbedPane3;
796  private javax.swing.JTable jTable1;
797  private javax.swing.JTable jTable2;
798  private javax.swing.JTextField jTextField1;
799  private javax.swing.JTextField jTextField2;
800  // End of variables declaration
801  private Conexion cn;
802  private Consultas cl;
803  private ResultSet resultado1, resultado2, resultado3, resultado4;
804  private DefaultTableModel dtm1, dtm2;
805  private JTableHeader th1, th2;
806
807  String filas1 [][] = {};
808  String columnas1 [] = {"Clave de personas", "Nombre"};
809
810  String filas2 [][] = {};
811  String columnas2 [] = {"Clave de mascotas", "Nombre", "Propietario"};
812

```


9. VentanaRegistroAdoptante

Este código permite llevar el registro de adoptantes del refugio de mascotas con sus respectivos datos. Se estableció conexión con la base de datos para validar y guardar los datos del adoptante.

```

1 package Mascotas;
2
3 import java.awt.Color;
4 import java.awt.Image;
5 import java.io.File;
6 import java.sql.SQLException;
7 import javax.swing.ImageIcon;
8 import javax.swing.JOptionPane;
9
10 public final class VentanaRegistroAdoptante extends javax.swing.JFrame {
11
12     public VentanaRegistroAdoptante() throws SQLException {
13         initComponents();
14
15         cn = new Conexion();
16         ldr = new LeerDatosRefugio();
17         gsp = new GetSetPersonas();
18         cl = new Consultas();
19
20         this.setLocationRelativeTo(null);
21         this.setConexion();
22
23         jTextField1.setEditable(false);
24
25         jTextField1.setText(String.valueOf(cl.getGeneroIdPersona()));
26
27         //Icono de la Ventana
28         try {
29             Image img = ImageIO.read(new File("calendar.png"));
30             this.setIconImage(img);
31         } catch (Exception e) {
32             JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
33         }
34     }
35
36     public void setConexion() {
37         if (cn.getConnection() != null) {
38             jLabel1.setText("Conectado");
39         }
40     }
41 }

```

```

45     }
46     else
47     {
48         jLabel17.setText("Desconectado");
49         jLabel17.setForeground(Color.red);
50     }
51 }
52
53
54 Generated Code
55
259
261 private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {
262     try
263     {
264         VentanaPrincipalRefugio vpv = new VentanaPrincipalRefugio();
265         this.dispose();
266         vpv.setVisible(true);
267     }
268     catch (SQLException e)
269     {
270         JOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
271     }
272 }
273
275 private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
276     System.exit(0);
277 }
278
279 private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
280     jTextField1.setText(null);
281     jTextField2.setText(null);
282     jTextField3.setText(null);
283     jTextField4.setText(null);
284     jComboBox1.setSelectedIndex(0);
285     jLabel18.setText(null);
286     jLabel19.setText(null);
287     jLabel110.setText(null);
288     jLabel111.setText(null);
289     jLabel112.setText(null);
290 }

```

```

292 private void jTextField2KeyTyped(java.awt.event.KeyEvent evt) {
293     if(!dv.getLeerNombre(jTextField2.getText()))
294         jLabel19.setText(null);
295     else
296         jLabel19.setText("El nombre es incorrecto");
297 }
298
299 private void jTextField3KeyTyped(java.awt.event.KeyEvent evt) {
300     if(jTextField3.getText().length() < 3)
301         jLabel111.setText("El domicilio es incorrecto");
302     else
303         jLabel111.setText(null);
304 }
305
306 private void jTextField4KeyTyped(java.awt.event.KeyEvent evt) {
307     char car = evt.getKeyChar();
308
309     if((car<'0' || car>'9'))
310         evt.consume();
311
312     if (jTextField4.getText().length()== 10)
313     {
314         evt.consume();
315         jLabel112.setText(null);
316     }
317     else
318         jLabel112.setText("El telefono es incorrecto");
319 }
320
321 private void jComboBox1ItemStateChanged(java.awt.event.ItemEvent evt) {
322     if(jComboBox1.getSelectedItem() == "Selecciona")
323         jLabel110.setText("Seleccione una opcion valida");
324     else
325         jLabel110.setText(null);
326 }
327
328 private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
329     try
330     {
331         gsp.setClave(Integer.parseInt(jTextField1.getText()));
332         gsp.setNombre(jTextField2.getText());
333         gsp.setGenero(jComboBox1.getSelectedItem().toString());
334         gsp.setDireccion(jTextField3.getText());
335         gsp.setTelefono(jTextField4.getText());
336     }
337     catch (Exception e)
338     {
339         JOptionPane.showMessageDialog(null,"Existe un error: "+e,"!! ERROR !!", 2);
340     }
341 }

```

```

335     }
336     catch (NumberFormatException e)
337     {
338     }
339
340
341     if (jTextField1.getText().length() == 0)
342     {
343         jLabel8.setText("La clave es incorrecta");
344     }
345     else
346     {
347         jLabel8.setText(null);
348
349         if (ldv.getLeerNombre(jTextField2.getText()))
350         {
351             jLabel9.setText(null);
352         }
353         else
354         {
355             jLabel9.setText("El nombre es incorrecto");
356
357             if (jTextField3.getText().length() > 3)
358             {
359                 jLabel11.setText(null);
360             }
361             else
362             {
363                 jLabel11.setText("El domicilio es incorrecto");
364
365                 if (jTextField4.getText().length() == 0)
366                 {
367                     jLabel12.setText(null);
368                 }
369                 else
370                 {
371                     jLabel12.setText("El telefono es incorrecto");
372
373                     if (jComboBox1.getSelectedItem() == "Selecciona")
374                     {
375                         jLabel10.setText("Selecciones una opcion valida");
376                     }
377                     else
378                     {
379                         jLabel10.setText(null);
380
381                         if (! (jComboBox1.getSelectedItem() != "Selecciona" & 0 <= jTextField1.getText().length() & ldv.getLeerNombre(gsp.getNombre() & ldv.getLeerTelefono(gsp.getTelefono())) || jTextField3.getText().length() <= 3)
382                             JOptionPane.showMessageDialog(null, "Los datos ingresados son incorrectos", "!! ERROR !!", 2);
383                     }
384                     else {
385                         try
386                         {
387                             if (cl.getLlenarPersonas(gsp.getClave(), gsp.getNombre(), gsp.getGenero(), gsp.getDireccion(), gsp.getTelefono()))
388                             {
389                                 JOptionPane.showMessageDialog(null, "Los datos han sido guardados", "Persona guardada", 1);
390
391                                 jTextField1.setText(String.valueOf(cl.getGenerarIdPersona()));
392                                 jTextField2.setText(null);
393
394                                 jTextField2.setText(null);
395                                 jTextField3.setText(null);
396                                 jTextField4.setText(null);
397                                 jComboBox1.setSelectedIndex(0);
398                                 jLabel8.setText(null);
399                                 jLabel9.setText(null);
400                                 jLabel10.setText(null);
401                                 jLabel11.setText(null);
402                                 jLabel12.setText(null);
403                             }
404                             else
405                             {
406                                 JOptionPane.showMessageDialog(null, "No se han guardado los datos", "!! ERROR !!", 2);
407                             }
408                         }
409                         catch (Exception e)
410                         {
411                             JOptionPane.showMessageDialog(null, "Existe un error: " + e, "!! ERROR !!", 2);
412                         }
413                     }
414                 }
415             }
416         }
417     }
418
419     private void jTextField2ActionPerformed(java.awt.event.ActionEvent evt) {
420
421     }
422
423     /**
424      * @param args the command line arguments
425      */
426     public static void main(String args[]) {
427
428         try {
429             for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
430                 if ("Nimbus".equals(info.getName())) {
431                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
432                     break;
433                 }
434             }
435         } catch (ClassNotFoundException ex) {
436             java.util.logging.Logger.getLogger(VentanaRegistroAdoptante.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
437         } catch (InstantiationException ex) {
438             java.util.logging.Logger.getLogger(VentanaRegistroAdoptante.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
439         } catch (IllegalAccessException ex) {
440             java.util.logging.Logger.getLogger(VentanaRegistroAdoptante.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
441         }
442     }

```

```

420         java.util.logging.Logger.getLogger(VentanaRegistroAdoptante.class.getName()).log(java.util.logging.Level.SEVERE,
421     } catch (javax.swing.UnsupportedLookAndFeelException ex) {
422         java.util.logging.Logger.getLogger(VentanaRegistroAdoptante.class.getName()).log(java.util.logging.Level.SEVERE,
423     }
424
425     java.awt.EventQueue.invokeLater(() -> {
426         try
427         {
428             new VentanaRegistroAdoptante().setVisible(true);
429         }
430         catch (Exception e)
431         {
432             JOptionPane.showMessageDialog(null, "Error: " + e, "!! ERROR !!", 2);
433         }
434     });
435 }

```

```

437 // Variables declaration - do not modify
438 private javax.swing.JButton jButton1;
439 private javax.swing.JComboBox jComboBox1;
440 private javax.swing.JLabel jLabel1;
441 private javax.swing.JLabel jLabel10;
442 private javax.swing.JLabel jLabel11;
443 private javax.swing.JLabel jLabel12;
444 private javax.swing.JLabel jLabel14;
445 private javax.swing.JLabel jLabel2;
446 private javax.swing.JLabel jLabel3;
447 private javax.swing.JLabel jLabel4;
448 private javax.swing.JLabel jLabel5;
449 private javax.swing.JLabel jLabel6;
450 private javax.swing.JLabel jLabel7;
451 private javax.swing.JLabel jLabel8;
452 private javax.swing.JLabel jLabel9;
453 private javax.swing.JMenu jMenu1;
454 private javax.swing.JMenu jMenu2;
455 private javax.swing.JMenuBar jMenuBar1;
456 private javax.swing.JMenuItem jMenuItem1;
457 private javax.swing.JMenuItem jMenuItem2;
458 private javax.swing.JMenuItem jMenuItem3;
459 private javax.swing.JPanel jPanel1;
460 private javax.swing.JPanel jPanel2;
461 private javax.swing.JTextField jTextField1;
462 private javax.swing.JTextField jTextField2;
463 private javax.swing.JTextField jTextField3;
464 private javax.swing.JTextField jTextField4;
465 // End of variables declaration
466 private Conexion cn;
467 private LeerDatosRefugio ldv;
468 private Consultas cl;
469 private GetSetPersonas gsp;
470 }

```

10. VentanaRegistroMascotas

Esta interfaz gráfica permite registrar mascotas en el refugio, se conecta a la base de datos y esto permite al usuario seleccionar un adoptante y registrar los detalles de la mascota para luego guardar los datos de manera segura.

```

1  package Mascotas;
2
3  import java.awt.Color;
4  import java.awt.Image;
5  import java.io.File;
6  import java.sql.SQLException;
7  import java.util.ArrayList;
8  import java.util.logging.Level;
9  import java.util.logging.Logger;
10 import javax.imageio.ImageIO;
11 import javax.swing.JOptionPane;
12

```



```

12
13 public class VentanaRegistroMascotas extends javax.swing.JFrame {
14
15     public VentanaRegistroMascotas() throws SQLException
16     {
17         initComponents();
18
19         cn = new Conexion();
20         ldv = new LeerDatosRefugio();
21         cl = new Consultas();
22         gsm = new GetSetMascotas();
23
24         lista = new ArrayList<String>();
25
26         this.setLocationRelativeTo(null);
27
28         jTextField1.setEditable(false);
29         jTextField4.setEditable(false);
30
31         jPanel13.setVisible(false);
32
33         this.setConexion();
34
35         lista = cl.getLlenarComboIdPersona();
36
37         jComboBox1.addItem("Selecciona");
38
39         for(int i = 0; i < lista.size(); i++)
40             jComboBox1.addItem(lista.get(i));
41
42         //Icono de la Ventana
43         try{
44             Image img = ImageIO.read(new File("calendar.png"));

```

```

43         try{
44             Image img = ImageIO.read(new File("calendar.png"));
45             this.setIconImage(img);
46         }catch (Exception e)
47         {
48             JOptionPane.showMessageDialog(null, "Existe un error: "+e, "!! ERROR !!", 2);
49         }
50     }
51
52     public void setConexion()
53     {
54         if(cn.getConectar() != null)
55         {
56             jLabel19.setText("Conectado");
57             //jLabel19.setForeground(Color.GREEN);
58         }
59         else
60         {
61             jLabel19.setText("Desconectado");
62             jLabel19.setForeground(Color.red);
63         }
64     }
65
66
67

```

Generated Code

```

329 private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
330     try
331     {
332         VentanaPrincipalRefugio vpv = new VentanaPrincipalRefugio();
333         this.dispose();
334         vpv.setVisible(true);
335     }
336     catch (SQLException e)
337     {
338         JOptionPane.showMessageDialog(null, "Existe un error: "+e, "!! ERROR !!", 2);
339     }
340 }
341
342 private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {
343     System.exit(0);
344 }
345

```

```

348 private void jMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
349     jComboBox1.setSelectedIndex(0);
350     jTextField1.setText(null);
351     jTextField2.setText(null);
352     jTextField3.setText(null);
353     jTextField4.setText(null);
354     jComboBox2.setSelectedIndex(0);
355     jComboBox3.setSelectedIndex(0);
356     jComboBox4.setSelectedIndex(0);
357     jLabel11.setText(null);
358     jLabel12.setText(null);
359     jLabel13.setText(null);
360     jLabel14.setText(null);
361     jLabel15.setText(null);
362     jLabel17.setText(null);
363 }
364
365 private void jTextField2KeyTyped(java.awt.event.KeyEvent evt) {
366     if(!dv.getLeerNombre(jTextField2.getText()))
367         jLabel11.setText(null);
368     else
369         jLabel11.setText("Nombre incorrecto");
370 }
371
372 private void jTextField3KeyTyped(java.awt.event.KeyEvent evt) {
373     char car = evt.getKeyChar();
374
375     if((car<'0' || car>'9'))
376         evt.consume();
377
378     if (jTextField3.getText().length() == 2)
379         evt.consume();
380
381     if(jTextField3.getText().length() > 0)
382         jLabel12.setText(null);
383 }
384 }

```

```

387 private void jComboBox2ItemStateChanged(java.awt.event.ItemEvent evt) {
388     if(jComboBox2.getSelectedItem() == "Selecciona")
389         jLabel13.setText("Opcion incorrecta");
390     else
391         jLabel13.setText(null);
392 }
393
394 private void jComboBox3ItemStateChanged(java.awt.event.ItemEvent evt) {
395     if(jComboBox3.getSelectedItem() == "Selecciona")
396         jLabel14.setText("Opcion incorrecta");
397     else
398         jLabel14.setText(null);
399 }
400
401 private void jComboBox4ItemStateChanged(java.awt.event.ItemEvent evt) {
402     if(jComboBox4.getSelectedItem() == "Selecciona")
403         jLabel15.setText("Opcion incorrecta");
404     else
405         jLabel15.setText(null);
406 }
407
408 private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {
409     try
410     {
411         if(jComboBox1.getSelectedItem() == "Selecciona")
412         {
413             jPanel3.setVisible(false);
414             jTextField1.setText(null);
415         }
416         else
417         {
418             jTextField4.setText(String.valueOf(cl.getGenerarIdMascota()));
419             jPanel3.setVisible(true);
420             jTextField1.setText(cl.getNombrePersona(Integer.parseInt((String) jComboBox1.getSelectedItem())));
421         }
422     }
423     catch (SQLException e)
424     {
425         JOptionPane.showMessageDialog(null, "Error: " + e, "!! ERROR !!", 2);
426     }
427 }

```

```

429 private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
430     try
431     {
432         gsm.setClave(Integer.parseInt(jTextField4.getText()));
433         gsm.setNombre(jTextField2.getText());
434         gsm.setEdad(Integer.parseInt(jTextField3.getText()));
435         gsm.setGenero(jComboBox2.getSelectedItem().toString());
436         gsm.setEspecie(jComboBox3.getSelectedItem().toString());
437         gsm.setEstatus(jComboBox4.getSelectedItem().toString());
438         gsm.setClavePrs(Integer.parseInt(jComboBox1.getSelectedItem().toString()));
439     }
440     catch (Exception e)
441     {
442     }
443
444     if (jTextField4.getText().length() == 0)
445     {
446         jLabel17.setText("Clave incorrecta");
447     }
448     else
449     {
450         jLabel17.setText(null);
451     }
452
453     if (ldv.getLeerNombre(jTextField2.getText()))
454     {
455         jLabel11.setText(null);
456     }
457     else
458     {
459         jLabel11.setText("Nombre incorrecto");
460     }
461
462     if (jTextField3.getText().length() >= 1)
463     {
464         jLabel12.setText(null);
465     }
466     else
467     {
468         jLabel12.setText("Edad incorrecta");
469     }
470
471     if (jComboBox2.getSelectedItem() == "Selecciona")
472     {
473         jLabel13.setText("Opcion incorrecta");
474     }
475     else
476     {
477         jLabel13.setText(null);
478     }
479
480     if (jComboBox3.getSelectedItem() == "Selecciona")
481     {
482         jLabel14.setText("Opcion incorrecta");
483     }
484     else
485     {
486         jLabel14.setText(null);
487     }
488
489     if (jComboBox4.getSelectedItem() == "Selecciona")
490     {
491         jLabel15.setText("Opcion incorrecta");
492     }
493     else
494     {
495         jLabel15.setText(null);
496     }
497 }

```

```

469 if (jComboBox4.getSelectedItem() == "Selecciona")
470 {
471     jLabel15.setText("Opcion incorrecta");
472 }
473 else
474 {
475     jLabel15.setText(null);
476 }
477
478 if (jTextField4.getText().length() > 0 & ldv.getLeerNombre(jTextField2.getText()) && jTextField3.getText().length() >
479     && jComboBox2.getSelectedItem() != "Selecciona" && jComboBox3.getSelectedItem() != "Selecciona" && jComboBox
480 {
481     if (cl.getLeerMascotas(gsm.getClavePrs(), gsm.getClave(), gsm.getNombre(), gsm.getEdad(),
482         gsm.getGenero(), gsm.getEspecie(), gsm.getEstatus()))
483     {
484         JOptionPane.showMessageDialog(null, "Los datos han sido guardado", "Mascota guardada", 1);
485
486         jComboBox1.setSelectedIndex(0);
487         jTextField1.setText(null);
488         jTextField2.setText(null);
489         jTextField3.setText(null);
490         jTextField4.setText(null);
491         jComboBox2.setSelectedIndex(0);
492         jComboBox3.setSelectedIndex(0);
493         jComboBox4.setSelectedIndex(0);
494         jLabel11.setText(null);
495         jLabel12.setText(null);
496         jLabel13.setText(null);
497         jLabel14.setText(null);
498         jLabel15.setText(null);
499         jLabel17.setText(null);
500     }
501     else
502     {
503         JOptionPane.showMessageDialog(null, "No se han guardado los datos", "!! ERROR !!", 2);
504     }
505 }
506 else
507 {
508     JOptionPane.showMessageDialog(null, "Los datos ingresados son incorrectos", "!! ERROR !!", 2);
509 }
510
511 /**
512  * @param args the command line arguments
513  */
514 public static void main(String args[]) {
515     try {
516         for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
517             if ("Nimbus".equals(info.getName())) {
518                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
519             }
520         }
521     }
522     catch (ClassNotFoundException ex) {
523         java.util.logging.Logger.getLogger(Main.class).log(java.util.logging.Level.SEVERE, null, ex);
524     }
525     catch (InstantiationException ex) {
526         java.util.logging.Logger.getLogger(Main.class).log(java.util.logging.Level.SEVERE, null, ex);
527     }
528     catch (IllegalAccessException ex) {
529         java.util.logging.Logger.getLogger(Main.class).log(java.util.logging.Level.SEVERE, null, ex);
530     }
531     finally {
532         new Main().setVisible(true);
533     }
534 }

```



```

511         if ("Nimbus".equals(info.getName())) {
512             javax.swing.UIManager.setLookAndFeel(info.getClassName());
513             break;
514         }
515     }
516     catch (ClassNotFoundException ex) {
517         java.util.logging.Logger.getLogger(VentanaRegistroMascotas.class.getName()).log(java.util.logging.Level.SEVERE,
518         catch (InstantiationException ex) {
519             java.util.logging.Logger.getLogger(VentanaRegistroMascotas.class.getName()).log(java.util.logging.Level.SEVERE,
520         catch (IllegalAccessException ex) {
521             java.util.logging.Logger.getLogger(VentanaRegistroMascotas.class.getName()).log(java.util.logging.Level.SEVERE,
522         catch (javax.swing.UnsupportedLookAndFeelException ex) {
523             java.util.logging.Logger.getLogger(VentanaRegistroMascotas.class.getName()).log(java.util.logging.Level.SEVERE,
524         }
525     }
526     //</editor-fold>
527
528     /* Create and display the form */
529     java.awt.EventQueue.invokeLater(new Runnable() {
530         public void run() {
531             try
532             {
533                 new VentanaRegistroMascotas().setVisible(true);
534             }
535             catch (SQLException e)
536             {
537                 JOptionPane.showMessageDialog(null, "Error: " + e, "!! ERROR !!", 2);
538             }
539         }
540     });

```

```

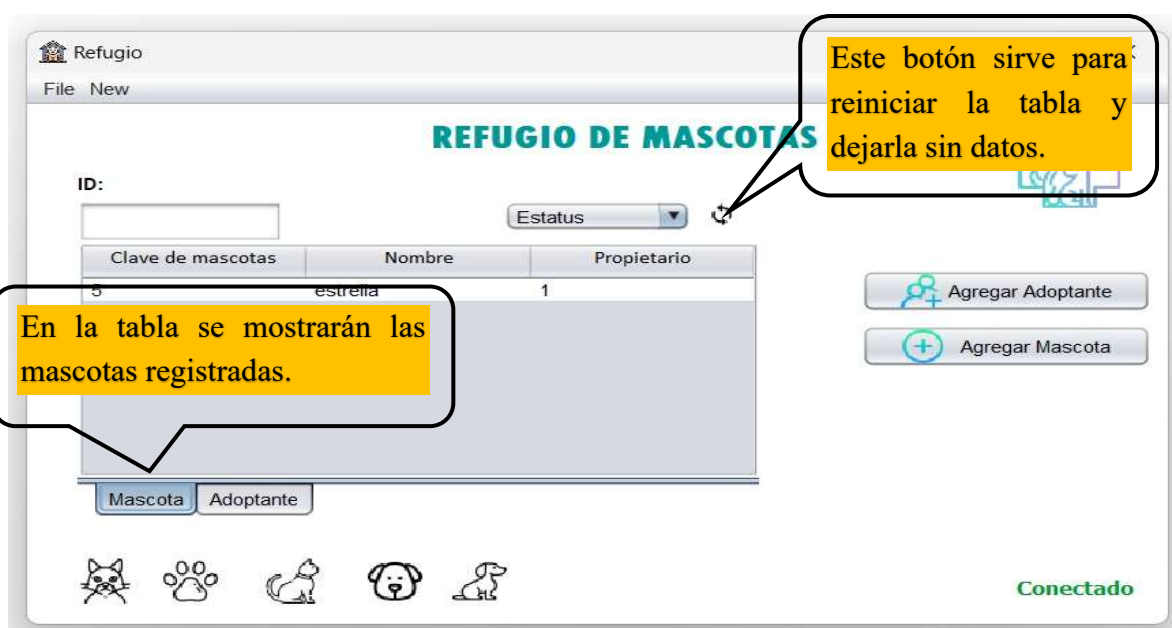
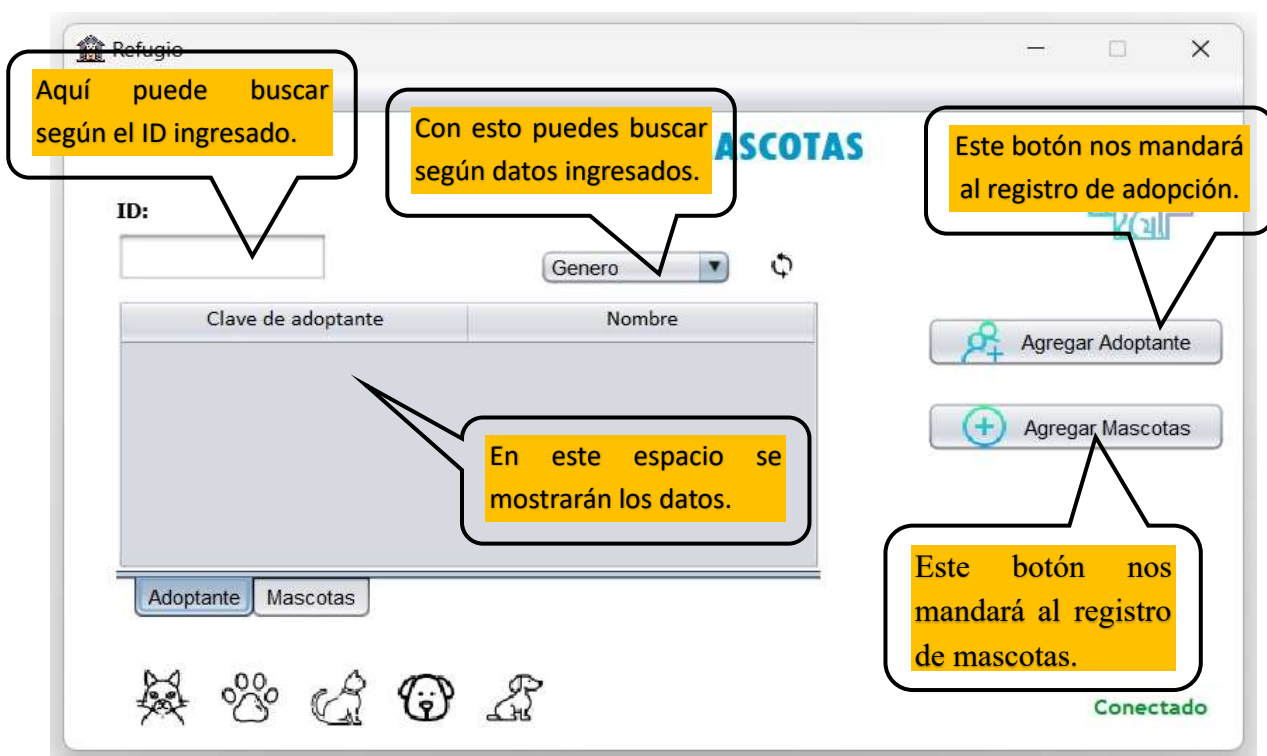
542     // Variables declaration - do not modify
543     private javax.swing.JButton jButton1;
544     private javax.swing.JComboBox jComboBox1;
545     private javax.swing.JComboBox jComboBox2;
546     private javax.swing.JComboBox jComboBox3;
547     private javax.swing.JComboBox jComboBox4;
548     private javax.swing.JLabel jLabel1;
549     private javax.swing.JLabel jLabel10;
550     private javax.swing.JLabel jLabel11;
551     private javax.swing.JLabel jLabel12;
552     private javax.swing.JLabel jLabel13;
553     private javax.swing.JLabel jLabel14;
554     private javax.swing.JLabel jLabel15;
555     private javax.swing.JLabel jLabel16;
556     private javax.swing.JLabel jLabel17;
557     private javax.swing.JLabel jLabel18;
558     private javax.swing.JLabel jLabel19;
559     private javax.swing.JLabel jLabel2;
560     private javax.swing.JLabel jLabel3;
561     private javax.swing.JLabel jLabel4;
562     private javax.swing.JLabel jLabel5;
563     private javax.swing.JLabel jLabel6;
564     private javax.swing.JLabel jLabel7;
565     private javax.swing.JLabel jLabel8;
566     private javax.swing.JLabel jLabel9;
567     private javax.swing.JMenu jMenu1;
568     private javax.swing.JMenu jMenu2;
569     private javax.swing.JMenuBar jMenuBar1;
570     private javax.swing.JMenuItem jMenuItem1;
571     private javax.swing.JMenuItem jMenuItem2;
572     private javax.swing.JMenuItem jMenuItem3;
573     private javax.swing.JPanel jPanel1;
574     private javax.swing.JPanel jPanel2;
575     private javax.swing.JPanel jPanel3;
576     private javax.swing.JTextField jTextField1;
577     private javax.swing.JTextField jTextField2;
578     private javax.swing.JTextField jTextField3;
579     private javax.swing.JTextField jTextField4;
580     // End of variables declaration
581     private Conexion cn;
582     private LeerDatosRefugio ldv;
583     private Consultas cl;
584     private GetSetMascotas gsm;
585     private ArrayList <String> lista;

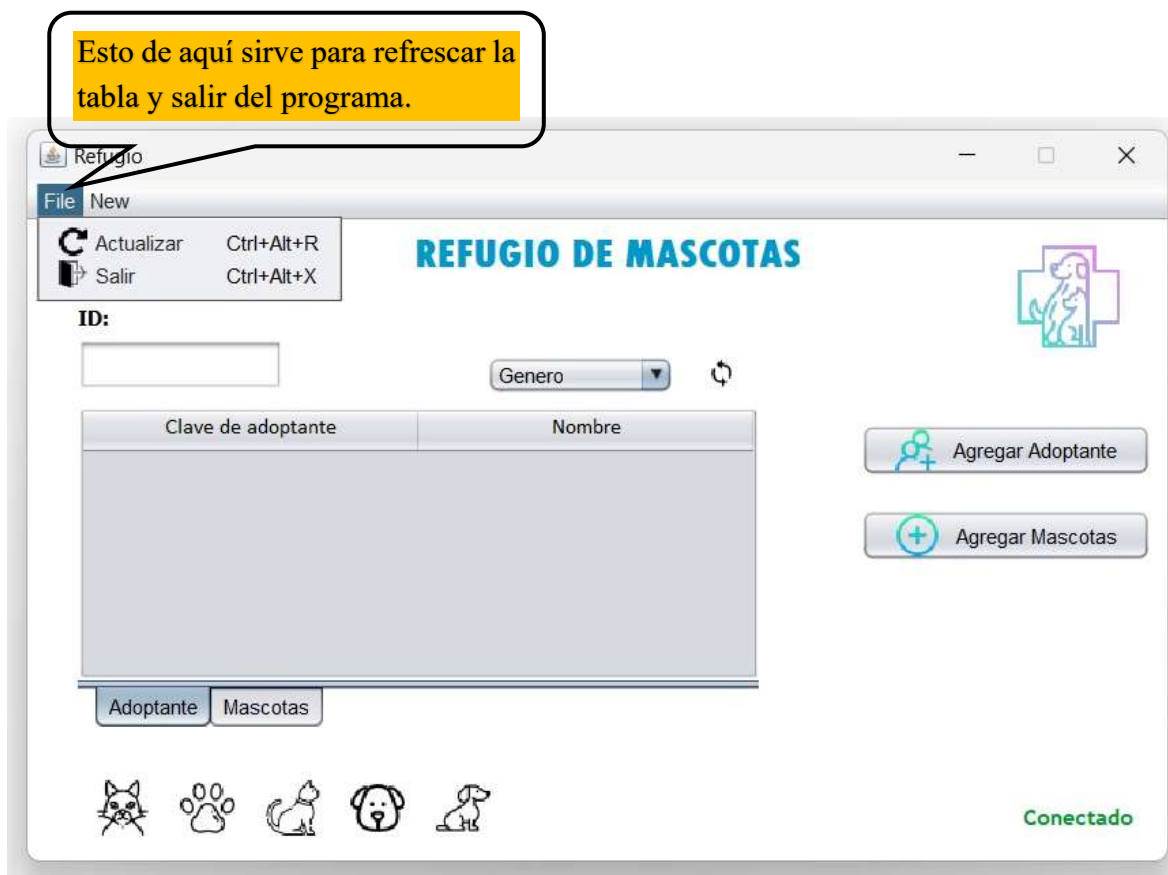
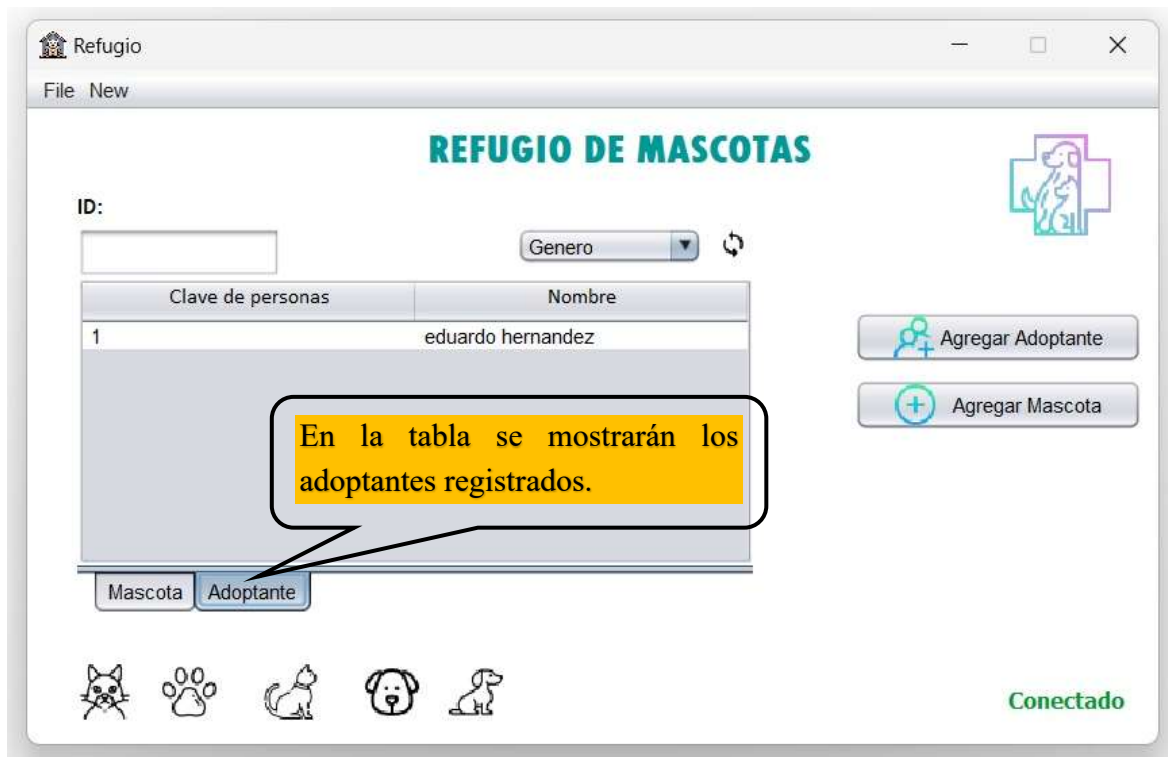
```

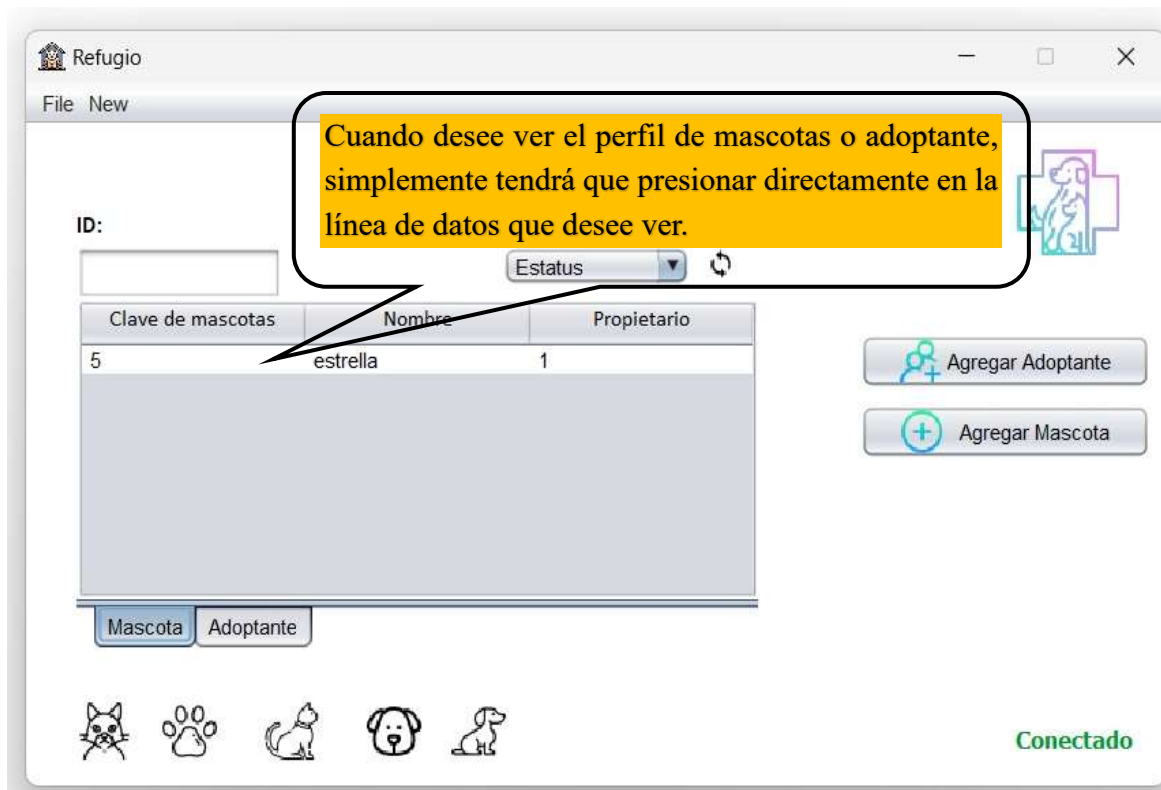
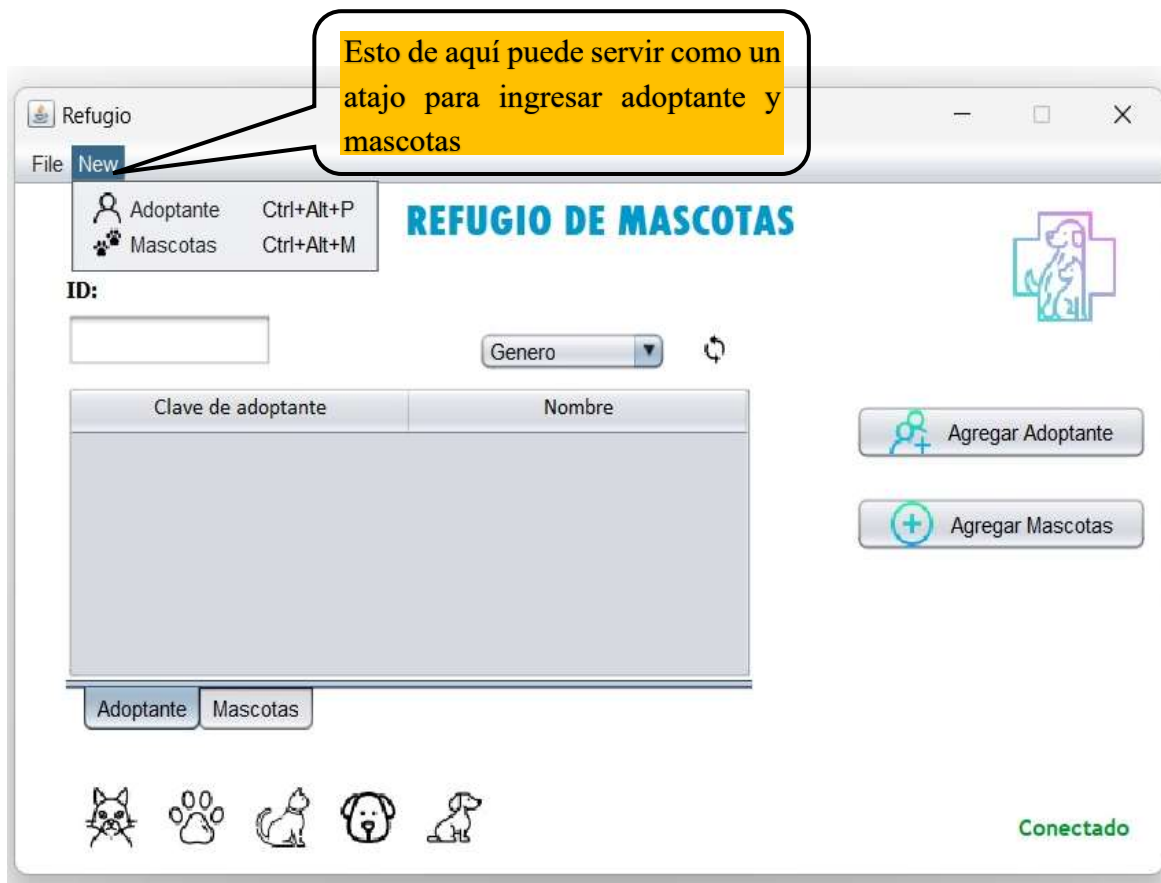
III. MANUAL DE USUARIO

1. Ventana Principal Refugio

Esta ventana funciona como el menú principal del programa donde encontraremos varias opciones.







2. Registrar Adoptante

Cuando use el botón de agregar adoptante en el menú principal, los mandará a una ventana donde tienen que ingresar los datos de las personas que deseen adoptar.

En estos espacios colocará los datos del adoptante.

REGISTRO ADOPTANTE

Clave: 1

Nombre: eduardo hernandez

Genero: Masculino

Domicilio: zona 5 huehuetenango

Telefono: 5024484838

Guardar

Conectado

Luego de completar los datos tienes que presionar aquí para guardarlos.

3. Registrar Mascota

Cuando use el botón de agregar mascotas en el menú principal, los mandará a la ventana donde tienen que ingresar los datos de las mascotas que estarán disponibles en el refugio.

En este espacio puede elegir a la persona que adoptará una mascota.

REGISTRO MASCOTAS

Adoptante: eduardo hernandez

Clave: 5

Nombre: estrella

Edad: 2

Genero: Hembra

Especie: Hamster

Estatus: Normal

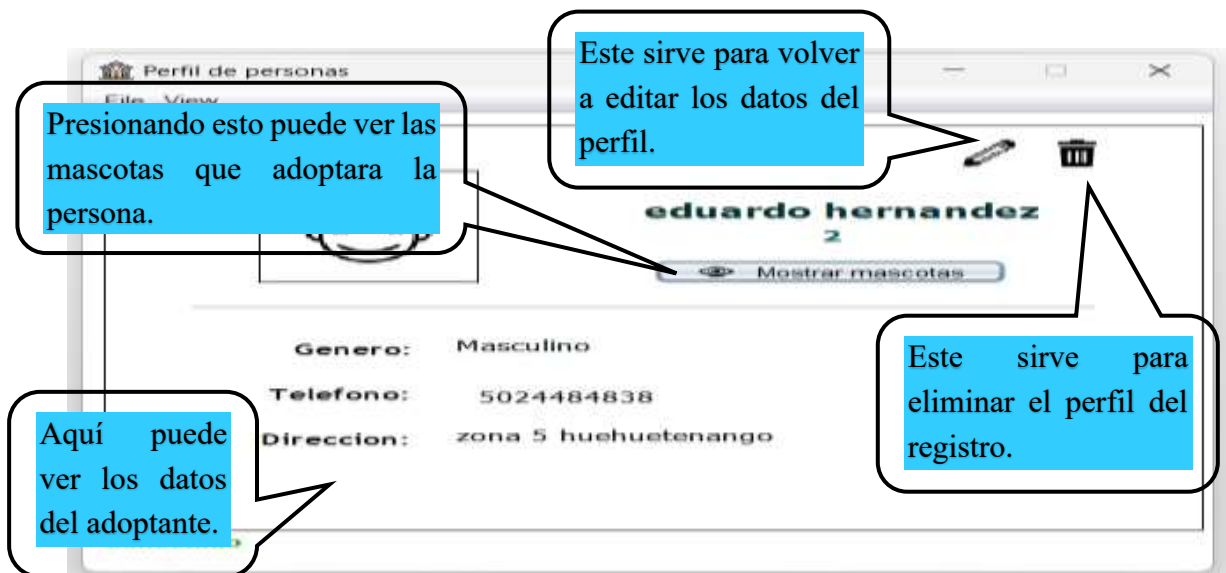
Guardar

En estos espacios colocará los datos de la mascota.

Presionando aquí se guardarán los datos.

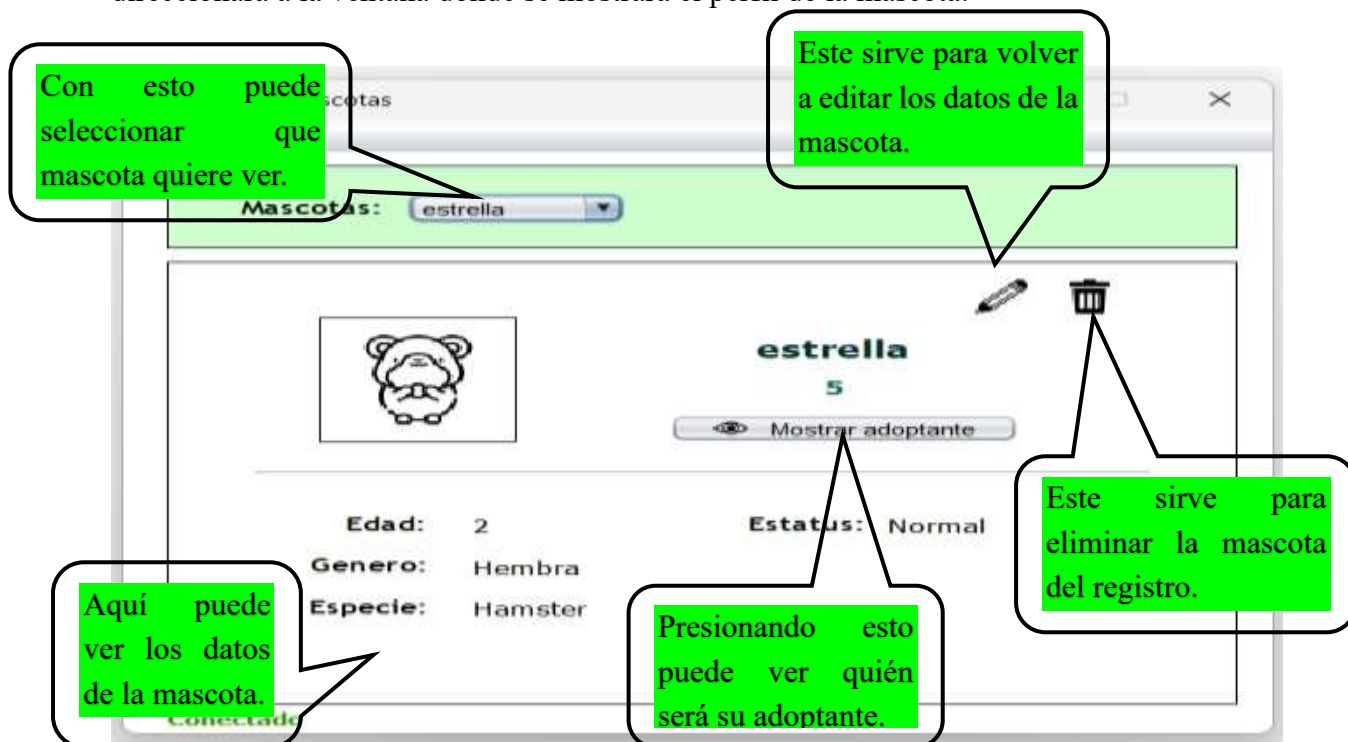
4. Mostrar Adoptante

Presionando directamente la línea de datos de la tabla adoptante en el menú principal te direccionará a la ventana donde se mostrará el perfil de la persona.



5. Mostrar Mascota

Presionando directamente la línea de datos de la tabla mascotas en el menú principal te direccionará a la ventana donde se mostrará el perfil de la mascota.



IV. CONCLUSIONES

- La colaboración entre todos los integrantes del grupo resultó en el desarrollo de un proyecto fluido y completo, donde cada miembro aportó algunas de sus habilidades en la elaboración de los códigos y a su vez esto permitió abordar los problemas desde diferentes ángulos y encontrar soluciones efectivas a los percances que se presentaron durante el desarrollo del proyecto del programa refugio de mascotas.
- El programa refugio de mascotas fue desarrollado con un diseño simple, pero que a su vez fuera un tanto alegre visualmente de manera que facilitara el uso del programa para la adopción de mascotas, pero esto también sirvió para simplificar el proceso de registro tanto de las personas que deseen adoptar como de las mascotas que ingresarían al refugio para su posterior adopción.
- Durante la elaboración del proyecto los integrantes aprendimos a tomar decisiones para poder cumplir con nuestra parte del proyecto que se nos fue asignada anteriormente y al mismo tiempo también supimos lo que es trabajar con plazos de tiempo establecidos para su entrega y unión de los códigos lo cual de cierta manera nos sirvió para practicar nuestras habilidades y poder mejorar nuestra comprensión de cómo podemos implementar códigos.
- En las últimas partes de la elaboración del programa pudimos aprender lo que es el desarrollo de base de datos, así como también como su implementación en las ramas del código para que estas se encargaran de guardar los datos ingresados y finalmente cuando se terminó el proyecto investigamos las maneras correctas de como subirlo a un repositorio en GitHub que creamos anteriormente.

V. REPOSITORIO EN GITHUB

<https://github.com/Alonzo-Morales/PROYECTO-GRUPO-5.git>

PD: Las carpetas se encuentran en la rama “máster”.

