



MANUAL TÉCNICO

“SISTEMA DE CONTROL DE ACCESO PARA EL INVENTARIO DEL ÁREA DE TI”

Versión 1.0

Fecha de publicación: 07 de enero de 2024



ÍNDICE DE IMÁGENES

Ilustración 1 Instalación de NetBeans	7
Ilustración 2 Opción de instalación de MYSQL	8
Ilustración 3 Opción MYSQL Server	8
Ilustración 4 Añadir MYSQL Workbench.....	9
Ilustración 5 MYSQL Shell	9
Ilustración 6 Requerimientos extras para MYSQL.....	10
Ilustración 7 Permisos de herramientas extras.....	10
Ilustración 8 Instalación de herramientas	11
Ilustración 9 Cuenta de MYSQL y contraseña	12
Ilustración 10 Inicio de sesión en Administrator	13
Ilustración 11 Importar o restaurar una base de datos	14
Ilustración 12 Selección de base de datos	15
Ilustración 13 Cargar base de datos.....	15
Ilustración 14 Base de datos.....	16
Ilustración 15 Agregar una librería	16
Ilustración 16 Librerías JAVA.....	17
Ilustración 17 Diseño de la ventana login	18
Ilustración 18 Lista de elementos en la ventana login	18
Ilustración 19 Importaciones de librerías JAVA usadas.....	19
Ilustración 20 Declaración de variables universales de login	20
Ilustración 21 Procesos iniciales de la ventana login	20
Ilustración 22 Configuración de cajas de texto y botón iniciar sesión	21
Ilustración 23 Acción del botón imprimir y su método.....	21
Ilustración 24 Método imprimir	22
Ilustración 25 Método imprimir segunda parte.....	22
Ilustración 26 Crear archivo PDF	23
Ilustración 27 Crear Archivo PDF parte 2.....	23
Ilustración 28 Crear archivo PDF parte 3	24
Ilustración 29 Validación de usuario.....	24
Ilustración 30 Validación de usuario 2 y envío de mensajes.....	25
Ilustración 31 Método validar usuario parte 3	26
Ilustración 32 Ventana Menú	27
Ilustración 33 Lista de elementos de la ventana Menú	27
Ilustración 34 Configuración de dimensiones de ventana Menú.....	28
Ilustración 35 Acción botón usuario de la ventana Menú	28
Ilustración 36 Diseño de la ventana Registro de Usuarios	29
Ilustración 37 Lista de elementos de la ventana Registrar Usuarios	29
Ilustración 38 Dimensiones de la ventana Registrar usuarios.....	30



Ilustración 39 Método actualizar de la ventana registrar usuarios	30
Ilustración 40 Método actualizar de la ventana menú parte 2	31
Ilustración 41 Método actualizar datos de la ventana menú parte 3	32
Ilustración 42 Opción de consulta de datos de un usuario	32
Ilustración 43 Método consultar de la ventana Registrar usuarios.....	33
Ilustración 44 Método consultar de la ventana Registrar usuarios parte 2	34
Ilustración 45 Método consultar de la ventana Registrar usuarios parte 3	35
Ilustración 46 Cambio de ventana Registrar usuarios a Consultar usuarios	35
Ilustración 47 Método consultar la información de un usuario.....	36
Ilustración 48 Método consultar la información de un usuario.....	36
Ilustración 49 Cambio a ventana menú y tipos de caracteres aceptados	37
Ilustración 50 Tipos de caracteres aceptados y ejecución del método actualizar	37
Ilustración 51 Guardar datos de un nuevo usuario.....	38
Ilustración 52 Guardar datos de un nuevo usuario parte 2	38
Ilustración 53 Diseño gráfico de la ventana de consulta de usuarios	39
Ilustración 54 Elementos de la ventana consultar usuarios.....	39
Ilustración 55 Programación de ventana consultar usuarios	40
Ilustración 56 Llamar los datos de todos los usuarios en la base de datos.....	40
Ilustración 57 Mandar datos a la tabla	41
Ilustración 58 Método eliminar usuario	41
Ilustración 59 Método actualizar	42
Ilustración 60 Método actualizar parte 2	42
Ilustración 61 Cambio a ventanas Menú, Registro de usuarios y acción al botón eliminar	43
Ilustración 62 Clase de conexión con la base de datos.....	43
Ilustración 63 Diseño preliminar del circuito electrónico	44
Ilustración 64 Programación de la placa Arduino	45
Ilustración 65 Ciclo loop para el accionamiento de puertas	46
Ilustración 66 Accionamiento hasta la puerta 7	47
Ilustración 67 Accionamiento hasta la puerta 10	48



CONTENIDO

INTRODUCCIÓN DEL SISTEMA.....	5
Objetivo del sistema	5
Descripción general del sistema	5
REQUERIMIENTOS DEL SISTEMA.....	6
Hardware	6
Software	6
INSTALACIÓN PREVIA DE HERRAMIENTAS.....	7
NetBeans	7
MYSQL	7
MYSQL Administrator	12
Arduino IDE.....	12
ARQUITECTURA DEL SISTEMA.....	13
Importación de base de datos en MYSQL Administrator	13
Adición de las librerías necesarias para el sistema	16
Diseño de ventana login y su programación.....	18
Diseño de ventana menu y su programación.	27
Diseño de ventana RegistrarUsuarios y su programación.....	29
Diseño de ventana ConsutaUsuarios y su programación.....	39
Programación de la clase ConexiónDB.	43
PROGRAMACIÓN EN ARDUINO.....	44
Diseño preliminar del circuito	44
Programación en Arduino IDE	45

INTRODUCCIÓN DEL SISTEMA

Objetivo del sistema

- Desarrollar un sistema de control de acceso para los equipos usados en el área de producción, mediante el uso de la metodología de prototipos, buscando mejorar la administración de los recursos informáticos en la organización.

Descripción general del sistema

- Crear un sistema que permita tener un control de acceso a los equipos informáticos resguardados, y al mismo tiempo registrar el préstamo de los dispositivos a cada uno de los empleados que tengan la necesidad de hacer uso de las tecnologías de la información; siendo administrado por parte del área de TI de la empresa AGROS S.A de C.V.

REQUERIMIENTOS DEL SISTEMA

Hardware

Los requerimientos mínimos de hardware con los que debe contar el equipo para poder hacer un uso del sistema de manera óptima, son los siguientes:

1. Computadora / Laptop con 4 GB de RAM, 250 GB de almacenamiento y Procesador Serie A (AMD) / Intel Celeron / Intel Pentium.
2. Relevadores de 5V (Según el numero de casilleros que se implementen).
3. Servomotor (variable según el material necesario en el diseño).
4. Chapas electronicas (Según el numero de casilleros que se implementen).
5. Paquete de Jumpers / cable 24-26 AWG
6. Estaño, cautín, pasta para soldar.
7. Tablillas de prueba.
8. Placas PCB (según el número de puertas y como se incorpore en las instalaciones).
9. Arduino UNO / Arduino MEGA y cable de alimentación USB (Según sea requerido por el número de casilleros).
10. Eliminador corriente (Voltaje recomendado de 12 V o como sea requerido por las chapas electrónicas).
11. Termoflit de distintas medidas y colores.

Software

Los requerimientos mínimos de software con los que debe contar el equipo para poder hacer un uso del sistema de manera óptima, son los siguientes:

1. 4 GB de RAM (PC / Laptop)
2. 250 GB de almacenamiento (PC / Laptop)
3. Procesador Serie A (AMD) / Intel Celeron / Intel Pentium (PC / Laptop)
4. NetBeans Versión 8.2.
5. Arduino IDE
6. MYSQL 8.0
7. MYSQL Administrator 8.0

INSTALACIÓN PREVIA DE HERRAMIENTAS

NetBeans

Realizamos la instalación sencilla de NetBeans manteniendo los datos que nos marca por default en cada uno de los apartados de instalación que nos muestra la herramienta de instalación. La ruta de instalación que genera de manera automática la tenemos en cuenta, solo mantenemos copiada la ruta de ubicación del programa para el caso de ser necesaria una reinstalación o desinstalación posterior.

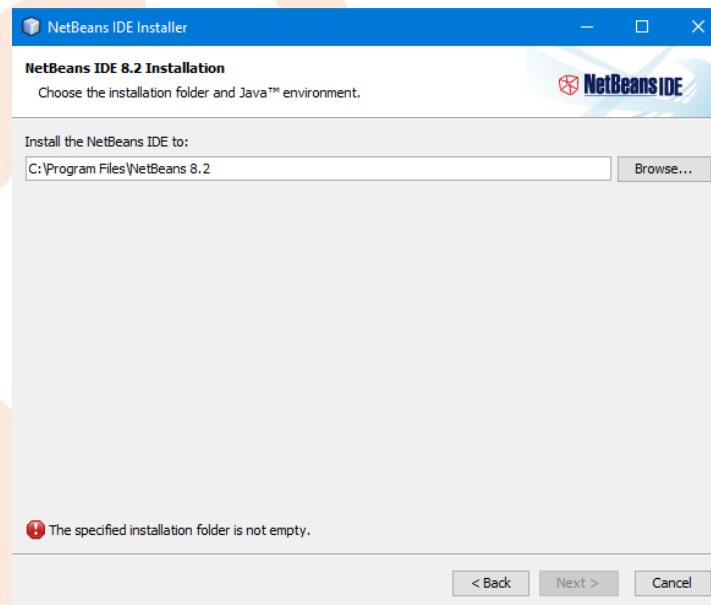


Ilustración 1 Instalación de NetBeans

MYSQL

Al comenzar la instalación de MYSQL se tienen varias opciones; en la imagen se observa la opción que se usa la forma “Custom” para solo instalar las herramientas que son necesarias en el sistema:

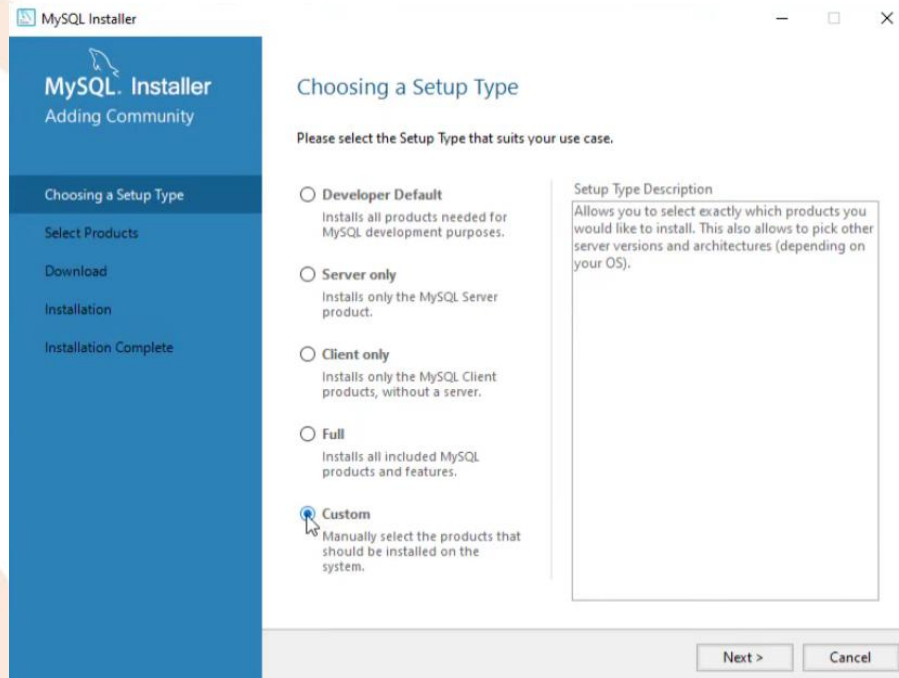


Ilustración 2 Opción de instalación de MYSQL

Agregamos las siguientes opciones, seleccionándolas y añadiéndolas con la flecha color verde:

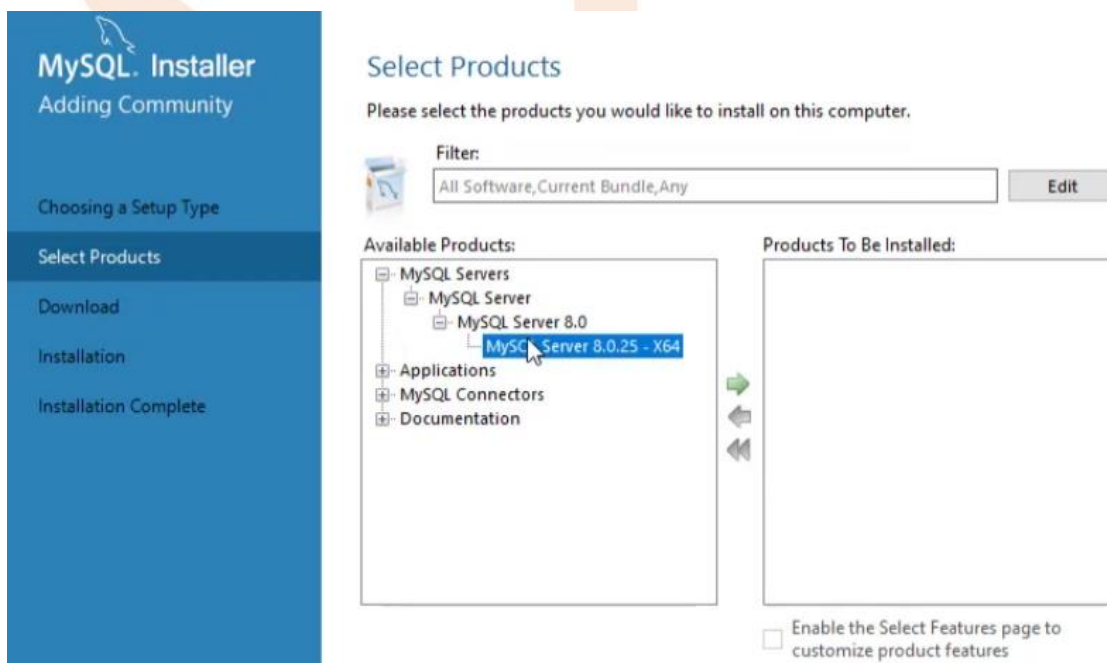


Ilustración 3 Opción MYSQL Server

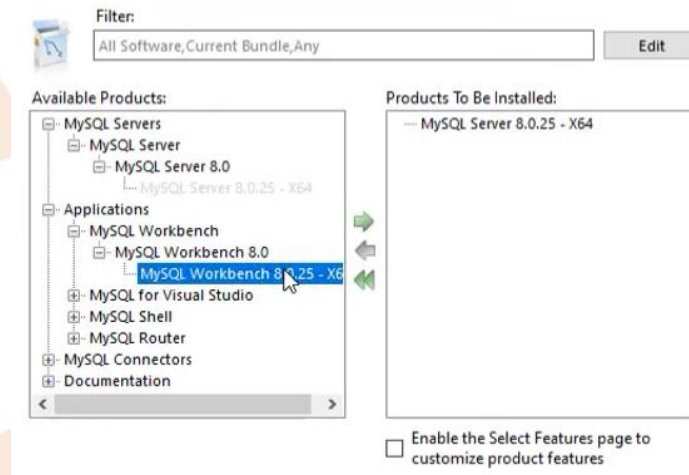


Ilustración 4 Añadir MYSQL Workbench

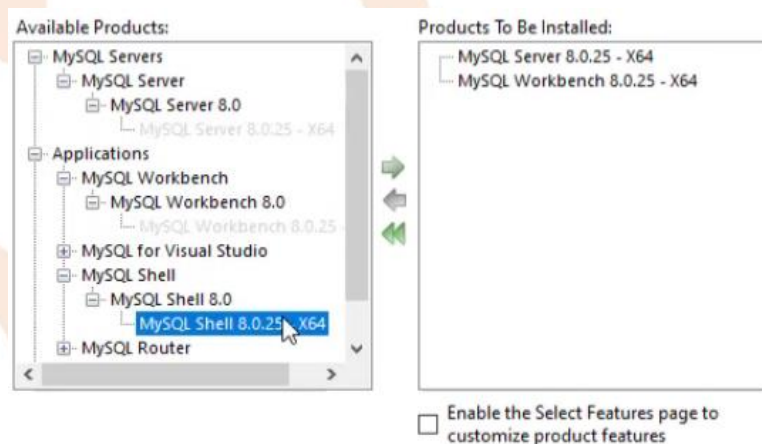


Ilustración 5 MYSQL Shell

Procede a seleccionar "NEXT" y en caso de ser la primera instalación de MYSQL, puede aparecer los siguientes requerimientos para la instalación:

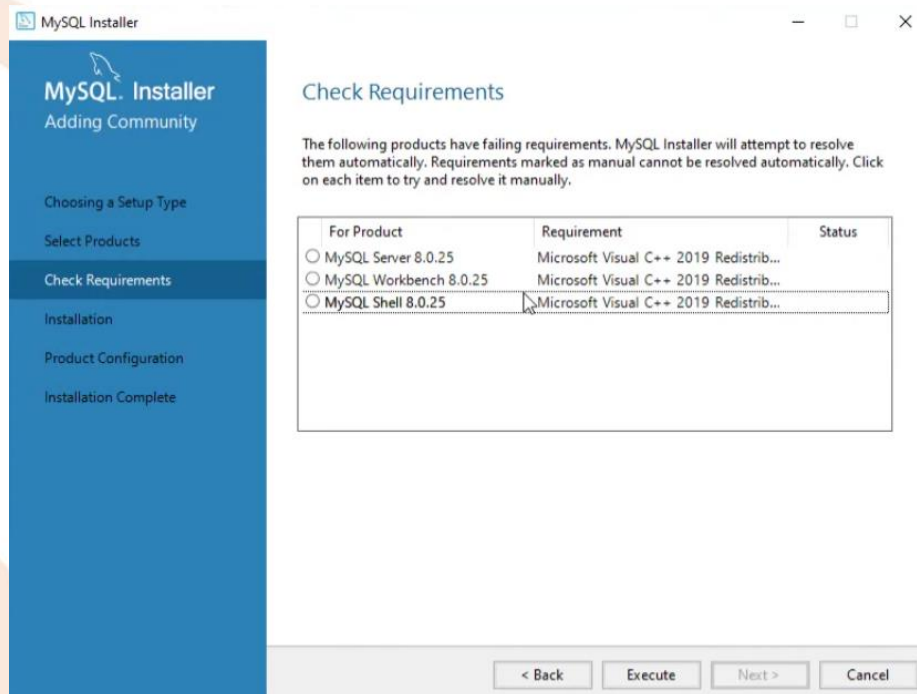


Ilustración 6 Requerimientos extras para MYSQL

Simplemente se da "Execute" para que se instale lo necesario. Para el caso anterior, aparecerán ventanas emergentes como la siguiente:

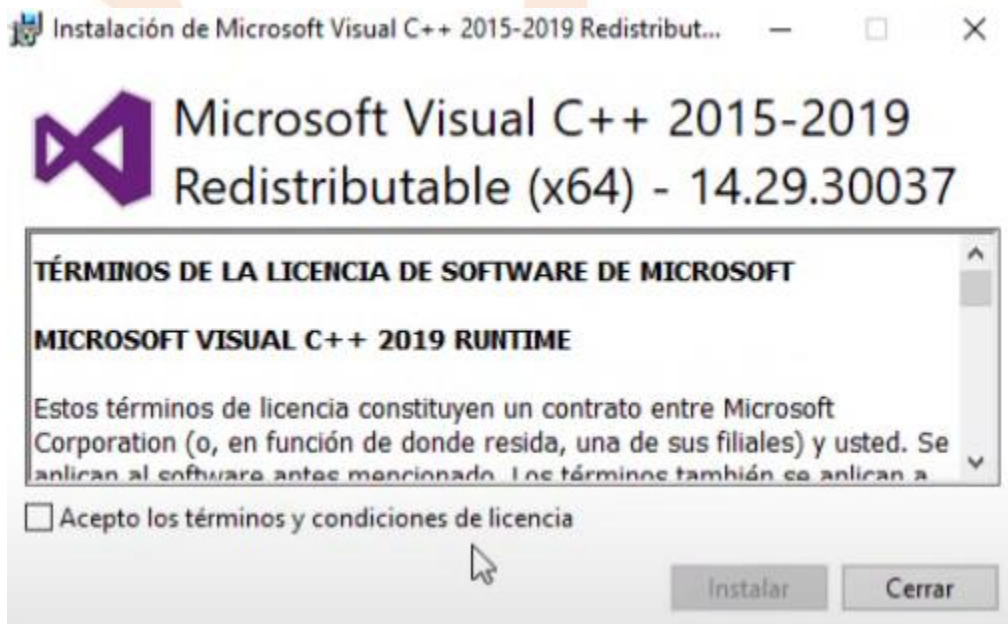


Ilustración 7 Permisos de herramientas extras

Simplemente se aceptan los términos necesarios y procede a presionar "Instalar", el procedimiento lo realizamos todas las ventanas emergentes que aparezcan en la pantalla. De esta forma realizamos una instalación correcta en cada una de las herramientas necesarias para MYSQL. Una vez terminada

la instalación de todas las herramientas extras, continuamos con la instalación de mysql que se muestra en la siguiente imagen:

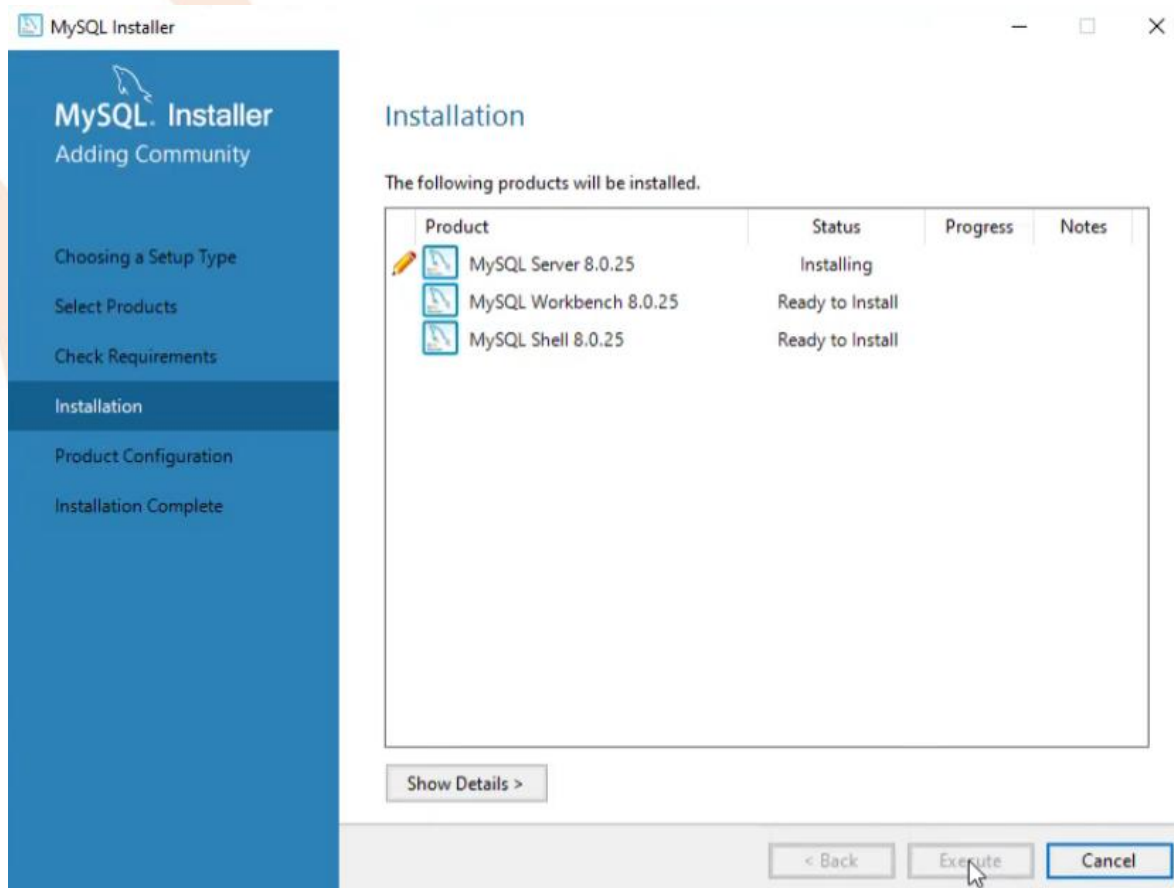


Ilustración 8 Instalación de herramientas

La ventana muestra las herramientas que se seleccionaron anteriormente y se procede a dar clic en el botón "Execute", de esta forma se proceden a instalar las herramientas necesarias para administrar nuestra base de datos. Al terminar el proceso, procedemos a dar clic en "Next".

A continuación, se muestra una ventana emergente donde los capos generados de manera automática se quedan de esa manera, lo único que debemos recordar es el número de puerto que se nos muestra de manera automática.

En la siguiente ventana encontramos la configuración que más nos importa que es la cuenta principal que utilizaremos en MYSQL, el usuario que colocaremos es "root" y la contraseña es "123root", siendo una forma fácil de recordar la contraseña.

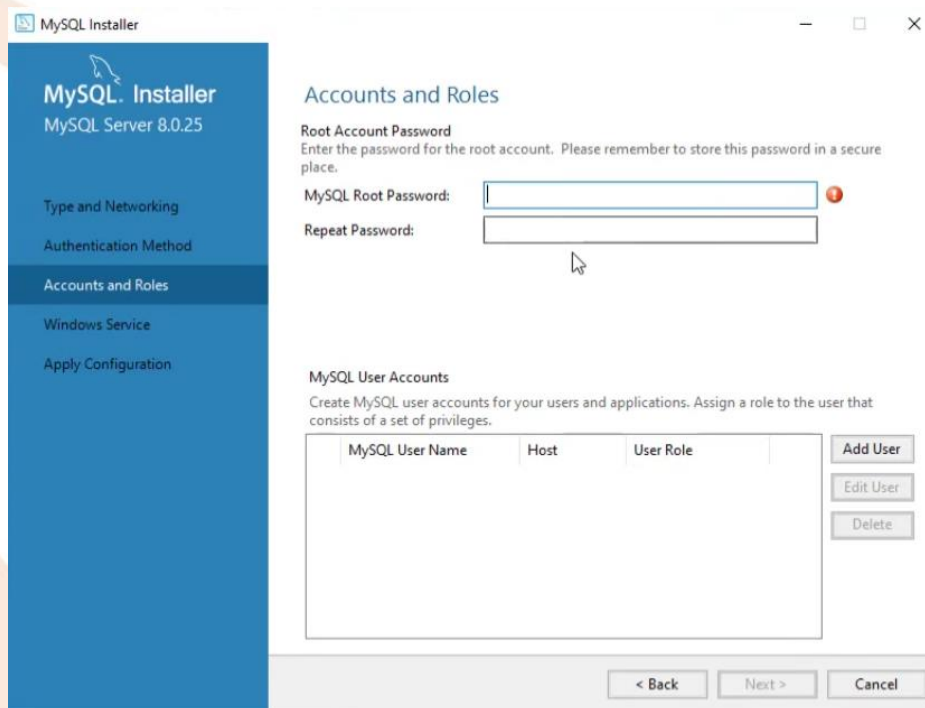


Ilustración 9 Cuenta de MYSQL y contraseña

Sigue adelante y las configuraciones posteriores se dejan de manera automática, cualquier programa que se inicialice después de terminar la instalación se debe cerrar.

MYSQL Administrator

Instalación de esta herramienta se hace al iniciar el instalador de la aplicación, toda la instalación se hace de manera convencional como en cualquier programa, mantenemos la configuración que se genera de manera automática.

Arduino IDE

La instalación de esta herramienta se realiza de manera convencional simplemente recordando la ruta donde se estarán generando los proyectos; esta ruta se puede verificar al momento de abrir el programa y guardar algún proyecto que se genere en esta aplicación

ARQUITECTURA DEL SISTEMA

Importación de base de datos en MYSQL Administrator

Iniciamos la aplicación MYSQL Administrator y realizamos el siguiente procedimiento para importar la base de datos que se usa en el sistema:

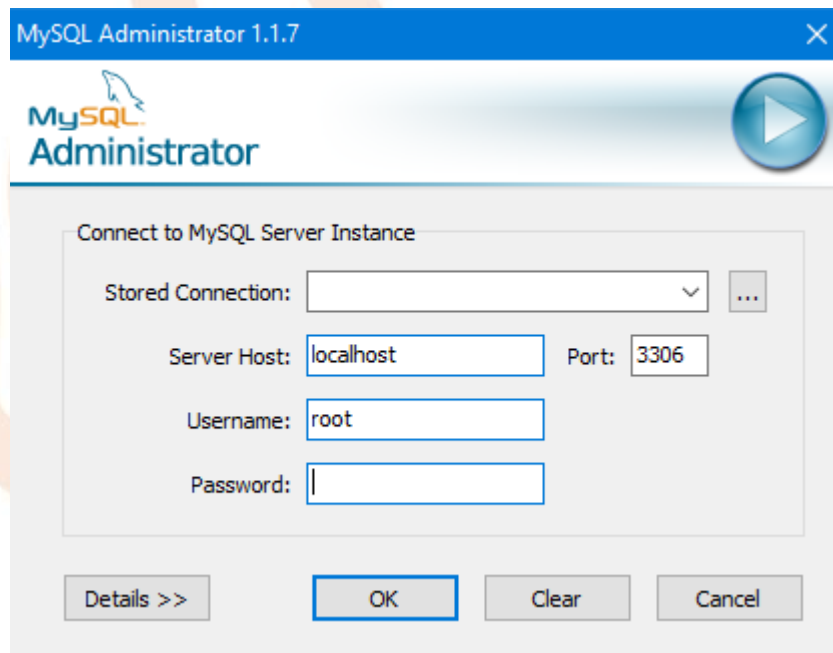


Ilustración 10 Inicio de sesión en Administrator

Ingresamos la contraseña que anteriormente se colocó en la configuración de la ilustración 9, y procedemos a tener acceso a la siguiente ventana y nos dirigimos a la siguiente selección del menú izquierdo como se observa en la imagen:

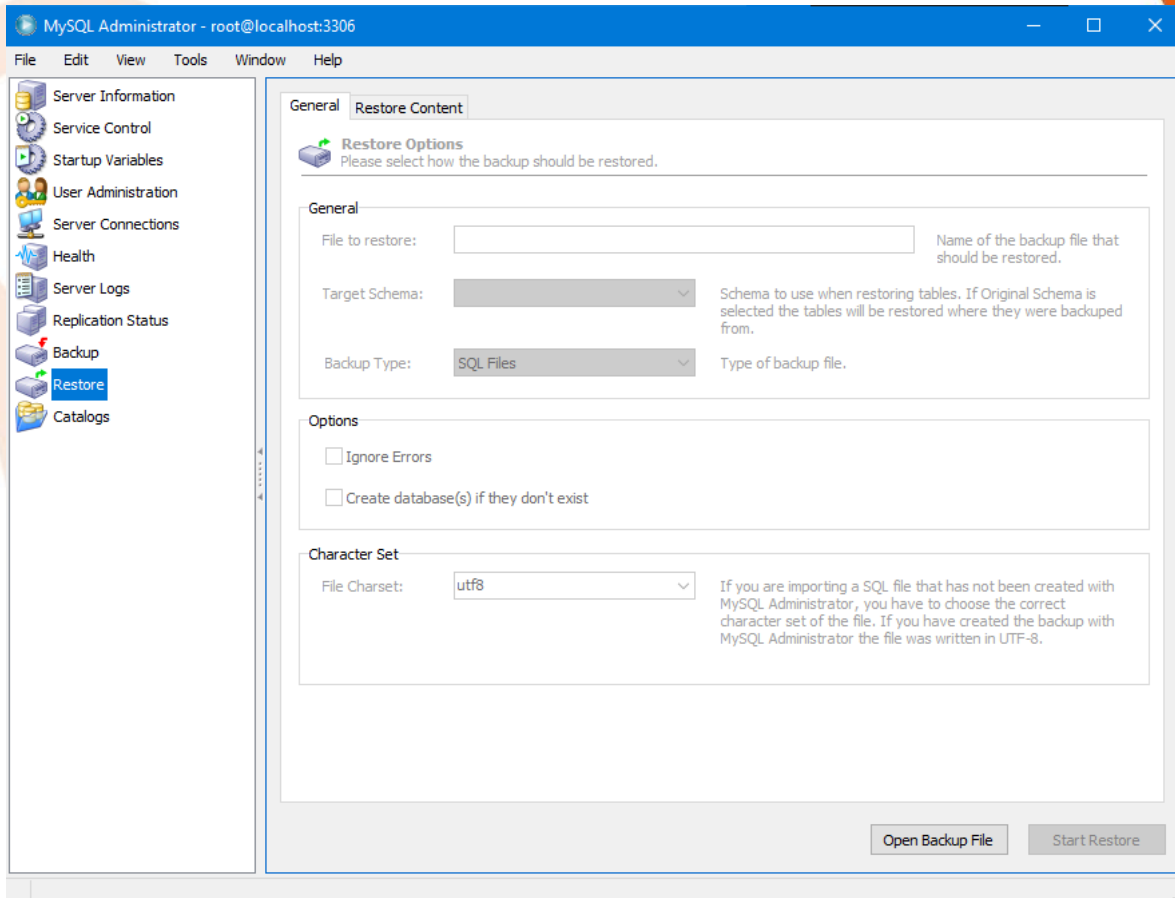


Ilustración 11 Importar o restaurar una base de datos

Seleccionamos el botón “Open Backup File” y nos dirigimos a la ruta donde tenemos la base de datos y la seleccionamos; como se muestra a continuación:

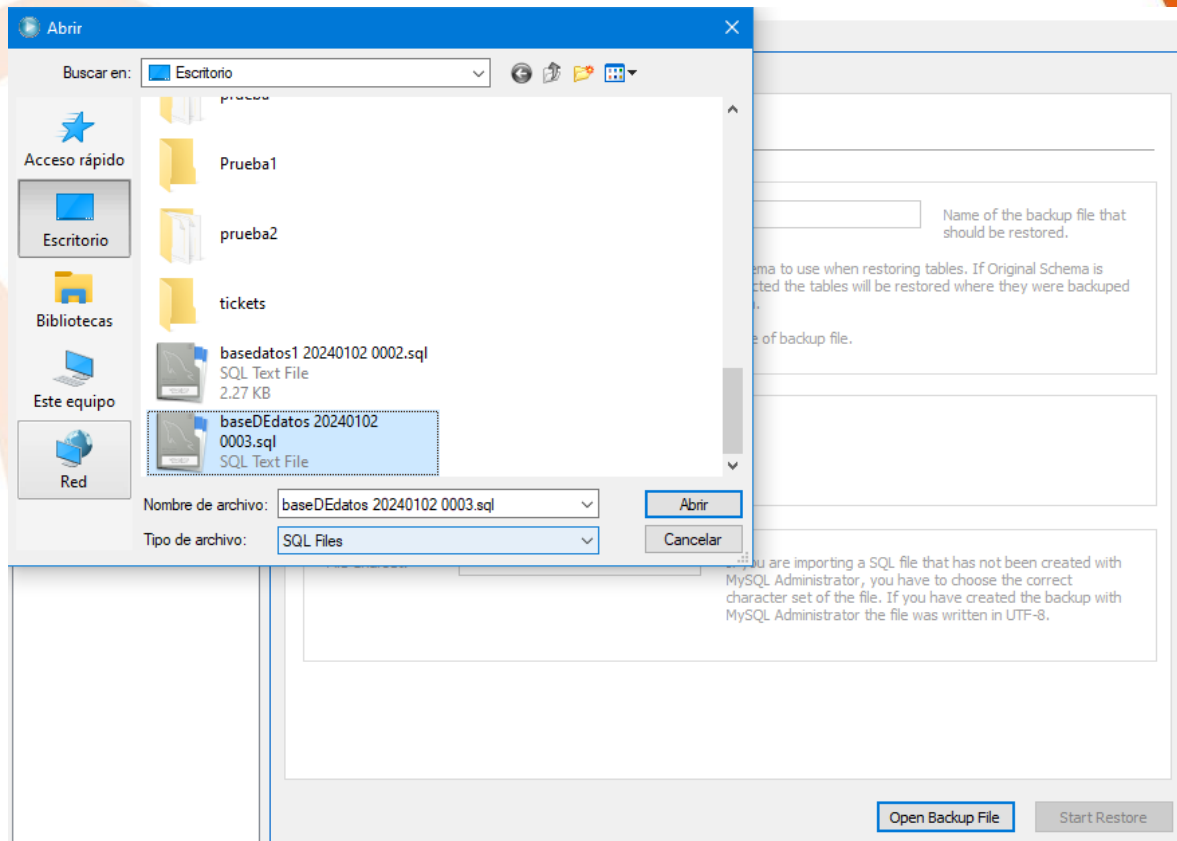


Ilustración 12 Selección de base de datos

Se muestra la siguiente configuración y se selecciona el botón “Star Restore”:

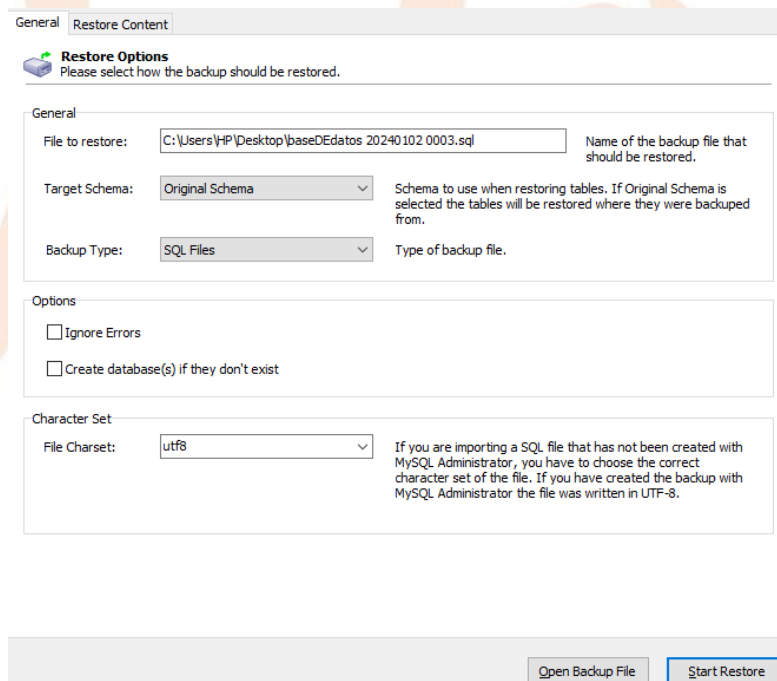


Ilustración 13 Cargar base de datos

Por último, nos vamos al apartado de “Catalogs” y nos muestra la siguiente base de datos llamada “controlAcceso”:

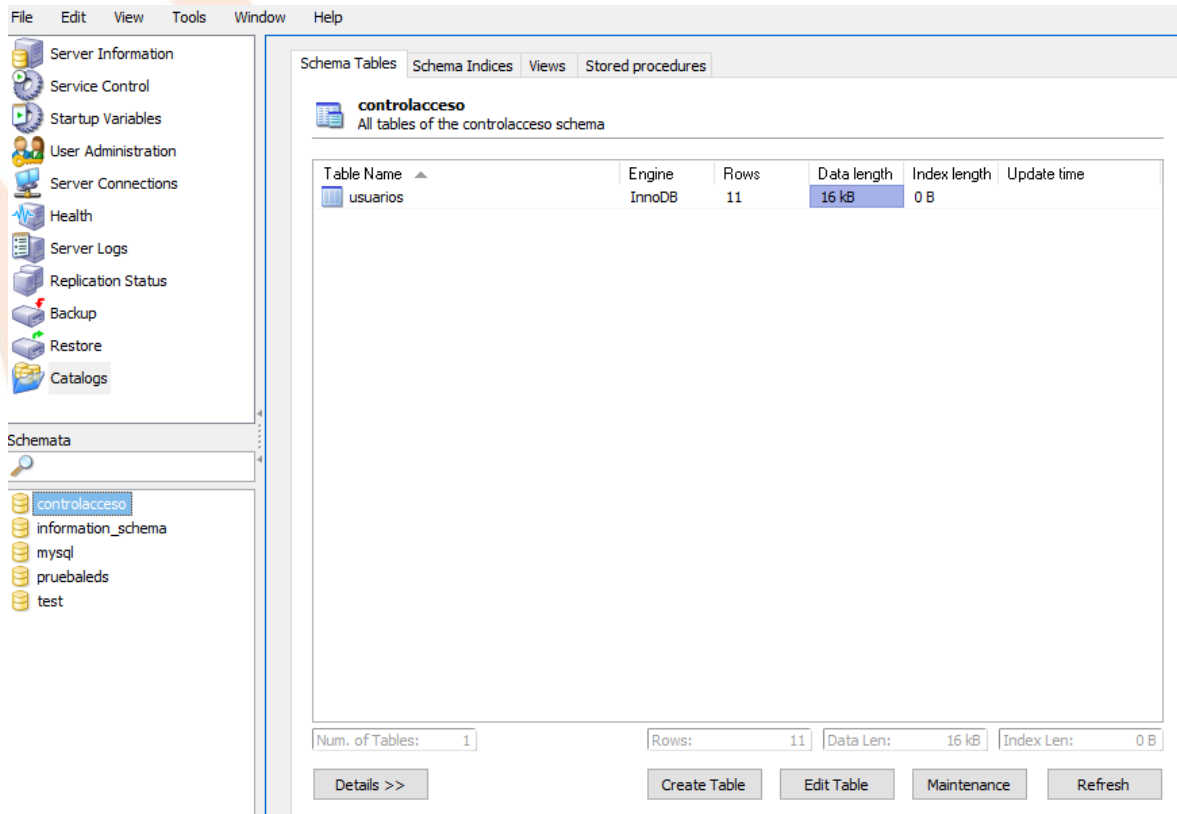


Ilustración 14 Base de datos

Adición de las librerías necesarias para el sistema

La forma en como añadir las librerías necesarias en el sistema, comienza dando clic derecho en el paquete “Libraries” ubicado en la ventana Projects, y seleccionando la opción “add JAR/Folder...” como se muestra de manera gráfica a continuación:

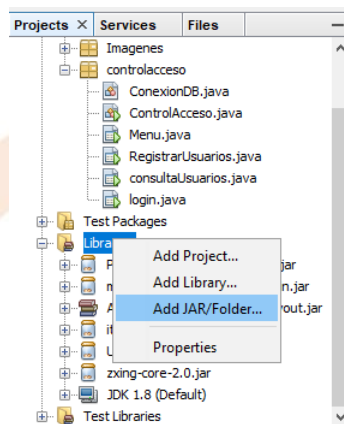


Ilustración 15 Agregar una librería

Aparece a continuación una ventana donde debemos buscar los archivos de las librerías que tienen la terminación “.jar”. Se tiene que ir agregando cada una por separado para evitar errores al cargar los datos de cada archivo.

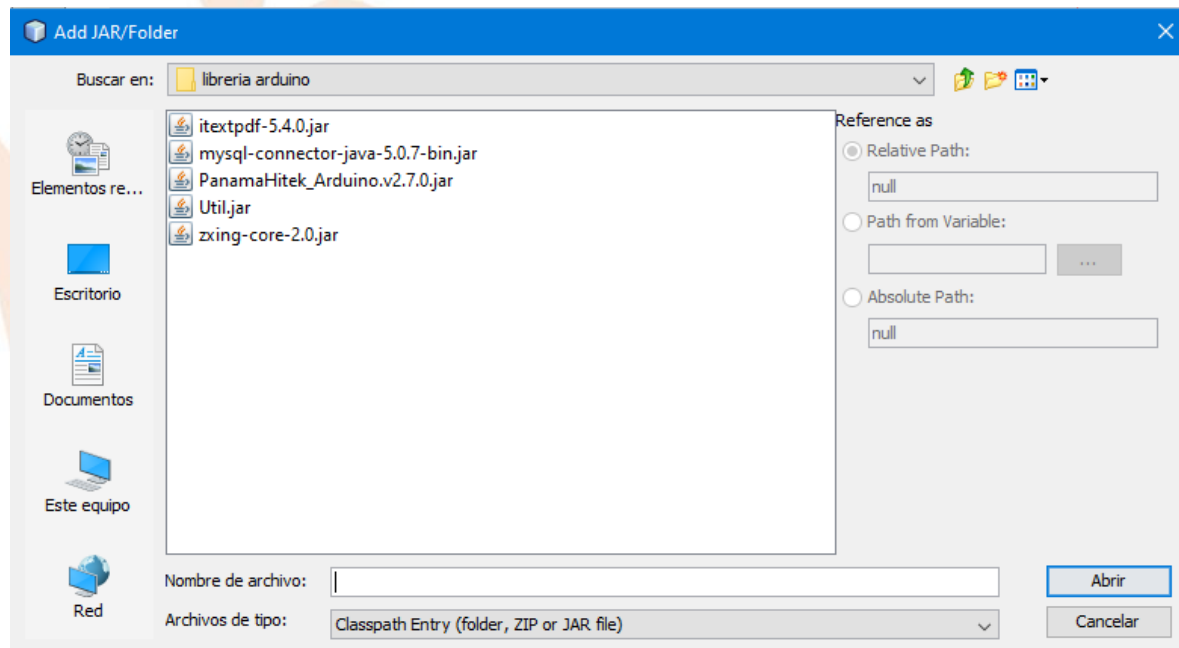


Ilustración 16 Librerías JAVA

Diseño de ventana login y su programación.

Encontramos en el diseño la siguiente configuración de los elementos añadidos a la ventana, junto a sus respectivos nombres:

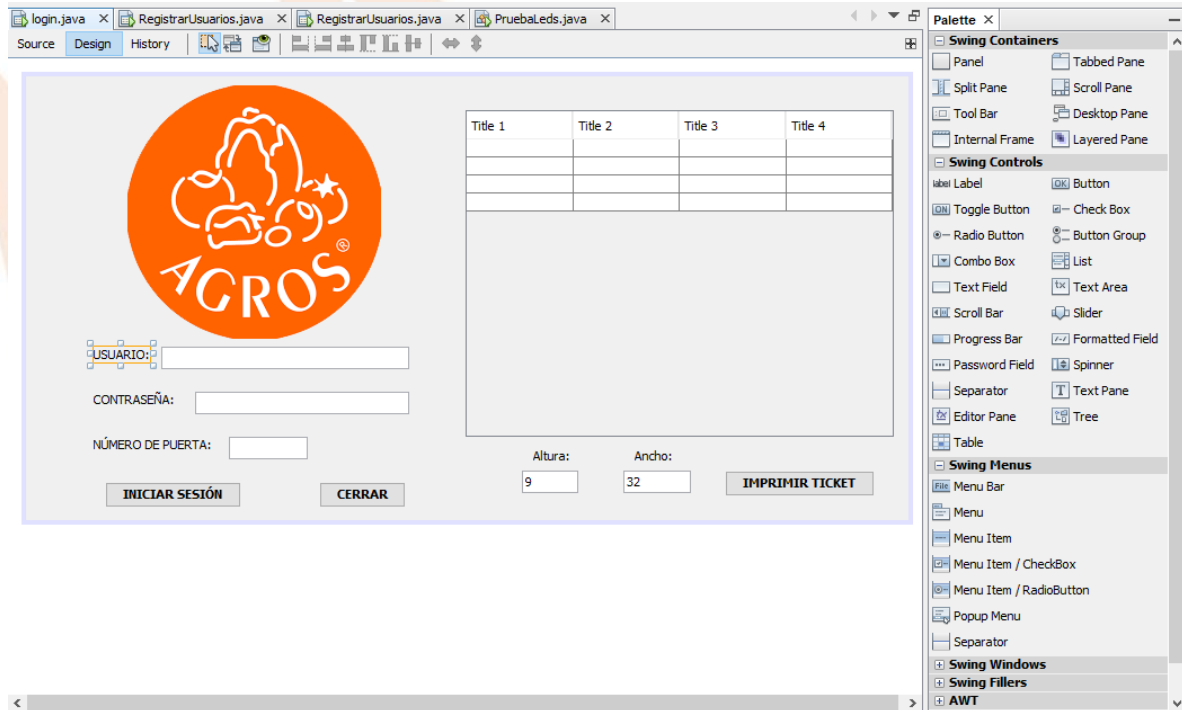


Ilustración 17 Diseño de la ventana login

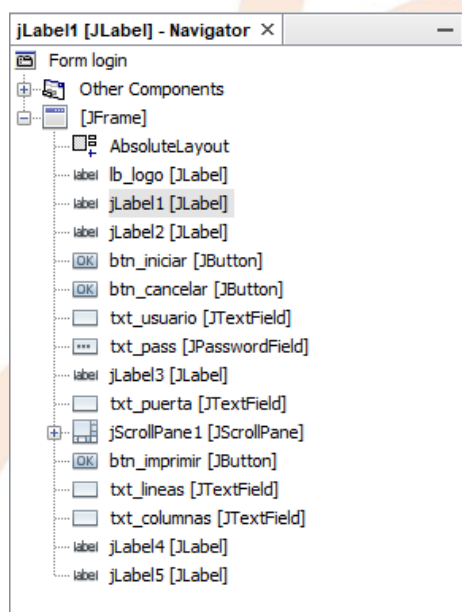



Ilustración 18 Lista de elementos en la ventana login

Comenzamos con el apartado de la programación en la interfaz de la ventana login, se muestra a continuación una imagen de referencia:



```

login.java x
Source Design History
2 import br.com.adilson.util.Extenso;
3 import br.com.adilson.util.PrinterMatrix;
4 import com.google.zxing.BarcodeFormat;
5 import com.google.zxing.WriterException;
6 import com.google.zxing.common.BitMatrix;
7 import com.google.zxing.qrcode.QRCodeWriter;
8 import com.itextpdf.text.Document;
9 import com.itextpdf.text.Image;
10 import com.itextpdf.text.Paragraph;
11 import com.itextpdf.text.pdf.PdfWriter;
12 import java.awt.Color;
13 import java.awt.Desktop;
14 import java.awt.Graphics2D;
15 import java.awt.Toolkit;
16 import java.awt.image.BufferedImage;
17 import java.io.*;
18 import java.sql.Connection;
19 import java.sql.ResultSet;
20 import java.sql.Statement;
21 import java.util.Calendar;
22 import java.util.GregorianCalendar;
23 import java.util.logging.Level;
24 import java.util.logging.Logger;
25 import javax.imageio.ImageIO;
26 import javax.print.Doc;
27 import javax.print.DocFlavor;
28 import javax.print.DocPrintJob;
29 import javax.print.PrintService;
30 import javax.print.PrintServiceLookup;
31 import javax.print.SimpleDoc;
32 import javax.print.attribute.HashPrintRequestAttributeSet;
33 import javax.print.attribute.PrintRequestAttributeSet;
34 import javax.swing.ImageIcon;
35 import javax.swing.JOptionPane;
36 import javax.swing.table.DefaultTableModel;
37 import panamahitek.Arduino.PanamaHitek_Arduino;
    
```

Ilustración 19 Importaciones de librerías JAVA usadas

Las importaciones de las librerías pueden ser colocadas de manera manual o por como se vaya pidiendo en la línea de código. Se continua con la declaración de variables universales a usar en esta ventana:



```

39 public class login extends javax.swing.JFrame {
40
41     DefaultTableModel registros = new DefaultTableModel();
42     ConexionDB cc = new ConexionDB();
43     Connection con = cc.conexion();
44     PanamaHitek_Arduino Arduino = new PanamaHitek_Arduino();
45     Calendar fecha = new GregorianCalendar();
46     String anho = Integer.toString(fecha.get(Calendar.YEAR));
47     String mes = Integer.toString(fecha.get(Calendar.MONTH));
48     String dia = Integer.toString(fecha.get(Calendar.DATE));
49     String fechacompleta = anho + "-" + (mes + 1) + "-" + dia;
50     String horas = Integer.toString(fecha.get(Calendar.HOUR_OF_DAY));
51     String minutos = Integer.toString(fecha.get(Calendar.MINUTE));
52     String horacompleta = horas + ":" + minutos;
53     String ruta_destino = null;
54     //VARIABLES SIN USO
55     //static String impresora = "EPSON L3210 Series";
56     //static String serial = "";
57     //static String grados = "0";
58

```

Ilustración 20 Declaración de variables universales de login

Prosigue la parte de dimensionamiento de la ventana, la llamada de imágenes disponible en el sistema y sentencias de títulos para la tabla de registro de actividad y la conexión con la placa Arduino:

```

58
59 public login() {
60     initComponents();
61     this.setTitle("LOGIN");
62     this.setSize(850, 450);
63     java.awt.Image img = Toolkit.getDefaultToolkit().getImage(getClass().getResource("/Imagenes/agros.png"));
64     this.setIconImage(img);
65     this.setLocation(270, 180);
66     String[] titulo = new String[]{"USUARIO", "FECHA", "HORA"};
67     registros.setColumnIdentifiers(titulo);
68     tabla_registros.setModel(registros);
69     try {
70         Arduino.arduinoTX("COM4", 9600);
71     } catch (Exception ex) {
72         Logger.getLogger(login.class.getName()).log(Level.SEVERE, null, ex);
73     }
74 }
75
76

```

Ilustración 21 Procesos iniciales de la ventana login

A continuación, se muestra la restricción de caracteres aceptados en los campos de escritura de algunas cajas de texto, al mismo tiempo, se observan los métodos a llamar en el botón "Iniciar sesión":



```

77 @SuppressWarnings("unchecked")
78 + Generated Code
192
193 - private void btn_cancelarActionPerformed(java.awt.event.ActionEvent evt) {
194     // Cerrar programa
195     System.exit(0);
196 }
197
198 - private void btn_iniciarActionPerformed(java.awt.event.ActionEvent evt) {
199     // Iniciar sesión
200     PDF();
201     ValidarUsuario();
202 }
203
204 - private void txt_usuarioKeyTyped(java.awt.event.KeyEvent evt) {
205     // TODO add your handling code here:
206     char c=evt.getKeyChar();
207     if((c<'a' || c>'z') && (c<'A' || c>'Z') && (c<' ' || c>' '))evt.consume();
208 }
209
210 - private void txt_puertaKeyTyped(java.awt.event.KeyEvent evt) {
211     // TODO add your handling code here:
212     char c=evt.getKeyChar();
213     if((c<'0' || c>'9'))evt.consume();
214 }
215
216 - private void txt_passKeyTyped(java.awt.event.KeyEvent evt) {
217     // TODO add your handling code here:
218     char c=evt.getKeyChar();
219     if((c<'0' || c>'9'))evt.consume();
220 }

```

Ilustración 22 Configuración de cajas de texto y botón iniciar sesión

Se tiene a continuación el siguiente apartado de acción en el botón imprimir y su respectivo método a llamar:

```

222 - private void btn_imprimirActionPerformed(java.awt.event.ActionEvent evt) {
223     //Evento imprimir
224     | Imprimir();
225     Imprimir2();
226 }
227
228 - private void Imprimir2(){
229     //Map<String, String> parametros = Map.of("nombrelPdf", "ticket.pdf", "impresora", impresora, "serial", serial, "grados", grados);
230
231     try {
232         File path = new File("ticket.pdf");
233         Desktop.getDesktop().open(path);
234
235         //C:\Users\HP\Documents\NetBeansProjects
236     } catch (IOException ex) {
237         Logger.getLogger(login.class.getName()).log(Level.SEVERE, null, ex);
238     }
239 }
240 }

```

Ilustración 23 Acción del botón imprimir y su método

El método llamado “Imprimir” que se observa a continuación es una función sin comprobar por la falta de una impresora de tickets, por lo cual no se realizaron pruebas en su configuración:

```

242 private void Imprimir() {
243     PrinterMatrix printer = new PrinterMatrix();
244
245     String archivo ="src/Imagenes/ticket.png";
246
247     Extenso e= new Extenso();
248     e.setNumber(101.85);
249
250     printer.setOutSize(9,32);
251     //printer.mapearDocumentoToFile(1, 1, archivo);
252     printer.mapearDocumento(1, 1, archivo);
253     printerToFile("ticket.txt");
254
255     FileInputStream inputStream = null;
256
257     try {
258         inputStream = new FileInputStream("ticket.pdf");
259     } catch (FileNotFoundException ex) {
260         ex.printStackTrace();
261     }
262
263     if (inputStream == null) {
264         return;
265     }
266
267     DocFlavor docFormat = DocFlavor.INPUT_STREAM.AUTODetect;
268     Doc document = new SimpleDoc(inputStream, docFormat, null);
269
270
271     PrintRequestAttributeSet attributeSet = new HashPrintRequestAttributeSet();
272     PrintService defaultPrintService = PrintServiceLookup.lookupDefaultPrintService();
273
274     if (defaultPrintService != null) {
275         DocPrintJob printJob = defaultPrintService.createPrintJob();
276         try {

```

Ilustración 24 Método imprimir

```

277         printJob.print(document, attributeSet);
278     } catch (Exception ex) {
279         System.out.println("Error: " + ex.toString());
280     }
281 }else{
282     JOptionPane.showMessageDialog(null, "No hay una impresora instalada");
283 }
284 }
285

```

Ilustración 25 Método imprimir segunda parte

Proseguimos con el método para generar un archivo PDF con los datos del usuario que intenta iniciar sesión en el sistema, tanto para la cuenta administrador y usuario normal.



```

286 private void PDF() {
287     int resultado = 0;
288     String usuario = txt_usuario.getText();
289     String pass = String.valueOf(txt_pass.getPassword());
290     String puerta = txt_puerta.getText();
291     String sql = "SELECT * FROM usuarios WHERE nombre='" + usuario + "' AND puerta='" + puerta + "' AND pass='" + pass + "'";
292
293     try {
294         Statement st = con.createStatement();
295         ResultSet rs = st.executeQuery(sql);
296         if (rs.next()) {
297             resultado = 1;
298             if (resultado == 1) {
299                 //CREAR QR
300                 String codigo = ("USUARIO: " + usuario + " | PUERTA: " + puerta + " | HORA: " + horacompleta + " | FECHA: " + fechacompleta).trim();
301                 //Tamaño del QR
302                 int size = 100;
303                 String FileType = "png";
304
305                 //CREAR nombre
306
307                 // Ruta de la imagen
308                 String filePath = "src/Imagenes/";
309                 //JFileChooser chooser = new JFileChooser();
310                 //chooser.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
311                 //if (chooser.showOpenDialog(this) == JFileChooser.APPROVE_OPTION) {
312                 //    filePath = chooser.getSelectedFile().getAbsolutePath();
313                 //}
314
315                 try {
316                     //GENERAR EL QR
317                     QRCodeWriter qrcode = new QRCodeWriter();
318                     BitMatrix matrix = qrcode.encode(codigo, BarcodeFormat.QR_CODE, size, size);
319                     File f = new File(filePath + "/ticket." + FileType);
320                     int matrixWidth = matrix.getWidth();
321                     BufferedImage image = new BufferedImage(matrixWidth, matrixWidth, BufferedImage.TYPE_INT_RGB);

```

Ilustración 26 Crear archivo PDF

```

322         image.createGraphics();
323
324         Graphics2D graphics2D = (Graphics2D) image.getGraphics();
325         graphics2D.setColor(Color.white); //color del fondo
326         graphics2D.fillRect(0, 0, matrixWidth, matrixWidth);
327         graphics2D.setColor(Color.black); //color QR
328
329         //GENERAR IMAGEN
330         for (int b = 0; b < matrixWidth; b++) {
331             for (int j = 0; j < matrixWidth; j++) {
332                 if (matrix.get(b, j)) {
333                     graphics2D.fillRect(b, j, 1, 1);
334                 }
335             }
336         }
337
338         //Mostrar imagen del QR
339         ImageIO.write(image, FileType, f);
340         java.awt.Image mImagen = new ImageIcon(filePath + "/ticket." + FileType).getImage();
341
342     } catch (WriterException ex) {
343         Logger.getLogger(login.class.getName()).log(Level.SEVERE, null, ex);
344     }
345
346     //Mandar a llamar la imagen de la empresa
347     Image header = Image.getInstance("src/Imagenes/agros.png");
348     Image header2 = Image.getInstance("src/Imagenes/ticket.png");
349     header.scaleToFit(50, 50);
350     //Crear PDF
351     FileOutputStream archivo = new FileOutputStream("ticket.pdf");
352     Document documento = new Document();
353     PdfWriter.getInstance(documento, archivo);
354     documento.open();
355
356     Paragraph parrafo = new Paragraph("PRESTAMO DE EQUIPOS INFORMÁTICOS");
357     documento.add(parrafo);
358     documento.add(header);

```

Ilustración 27 Crear Archivo PDF parte 2

```

359 documento.add(new Paragraph("USUARIO: " + usuario));
360 documento.add(new Paragraph("PUERTA: " + puerta));
361 documento.add(new Paragraph("FECHA: " + fechacompleta));
362 documento.add(new Paragraph("HORA: " + horacompleta));
363 documento.add(header2);
364 documento.close();
365
366 //File path = new File("ticket.pdf");
367 //Desktop.getDesktop().open(path);
368 }
369 }
370 } catch (Exception e) {
371     JOptionPane.showMessageDialog(null, "NO EXISTE EL USUARIO");
372 }
373
374 }
    
```

Ilustración 28 Crear archivo PDF parte 3

Prosigue la creación del método de validación de usuario, que se encarga de hacer una búsqueda en la base de datos, para corroborar que los datos que ingresan, efectivamente, están registrados en la base de datos. Se tiene 2 tipos de acciones, siendo una para la cuenta *administrador* y la otra para la cuenta de *usuario normal*.

```

376 private void ValidarUsuario() {
377     int resultado = 0;
378     String usuario = txt_usuario.getText();
379     String pass = String.valueOf(txt_pass.getPassword());
380     String puerta = txt_puerta.getText();
381     File archivo;
382     FileWriter escribir;
383     PrintWriter linea;
384     archivo = new File("Registros" + fechacompleta + ".txt");
385
386     int p = Integer.valueOf(puerta);
387
388     String sql = "SELECT * FROM usuarios WHERE nombre='" + usuario + "' AND puerta='" + puerta + "' AND pass='" + pass + "'";
389
390     try {
391         Statement st = con.createStatement();
392         ResultSet rs = st.executeQuery(sql);
393         if (rs.next()) {
394             resultado = 1;
395             if (resultado == 1 && p == 0) {
396                 Menu cu = new Menu();
397                 cu.setVisible(true);
398                 try {
399                     registros.addRow(new Object[]{
400                         usuario, fechacompleta, horacompleta
401                     });
402
403                     if (!archivo.exists()) {
404                         try {
405                             archivo.createNewFile();
406                             escribir = new FileWriter(archivo, true);
407                             linea = new PrintWriter(escribir);
408                             linea.print(usuario + "|");
409                             linea.print(fechacompleta + "|");
410                             linea.println(horacompleta);
411                             linea.close();
412                         } catch (Exception e) {
    
```

Ilustración 29 Validación de usuario

Después de verificar que el usuario si está registrado, en el caso de ser un *usuario normal*, procede a enviar el número de puerta que se tiene asignado al *usuario normal* hacia la placa Arduino, para

que en su programación se accionen los componentes correspondientes a ese número de puerta. En caso de ser una cuenta *Administrador* se redirecciona a la ventana de menú.

```

413         } catch (Exception e) {
414             JOptionPane.showMessageDialog(null, "Error al crear el Registros.txt");
415         }
416     } else {
417         try {
418             escribir = new FileWriter(archivo, true);
419             linea = new PrintWriter(escribir);
420             linea.print(usuario + "|");
421             linea.print(fechacompleta + "|");
422             linea.println(horacompleta);
423             linea.close();
424         } catch (Exception e) {
425             JOptionPane.showMessageDialog(null, "Error al crear el Registros.txt");
426         }
427     }
428     } catch (Exception e) {
429     }
430     this.dispose();
431 } else {
432     try {
433         Arduino.sendData(puerta);
434     }
435     try {
436         registros.addRow(new Object[]{
437             usuario, fechacompleta, horacompleta
438         });
439     }
440     if (!archivo.exists()) {
441         try {
442             //Crear archivo txt
443             archivo.createNewFile();
444             escribir = new FileWriter(archivo, true);
445             linea = new PrintWriter(escribir);
446             linea.print(usuario + "|");
447             linea.print(fechacompleta + "|");
448             linea.println(horacompleta);

```

Ilustración 30 Validación de usuario 2 y envío de mensajes

En este mismo método, se tiene el proceso de generar un historial en un archivo "Registro.txt" que se guarda en la carpeta principal del proyecto, donde se almacenan los datos de nombre de usuario, su número de puerta, la fecha y la hora en la que tiene acceso; se muestran todos los ingresos correctos que se generaron en el sistema.

```

449         linea.close();
450
451         } catch (Exception e) {
452             JOptionPane.showMessageDialog(null, "Error al crear el Registros.txt");
453         }
454     } else {
455         try {
456
457             //Reescritción del archivo de registros
458             escribir = new FileWriter(archivo, true);
459             linea = new PrintWriter(escribir);
460             linea.print(usuario + "|");
461             linea.print(fechacompleta + "|");
462             linea.println(horacompleta);
463             linea.close();
464         } catch (Exception e) {
465             JOptionPane.showMessageDialog(null, "Error al crear el Registros.txt");
466         }
467     }
468     } catch (Exception e) {
469     }
470     System.out.println("Se tuvo acceso a la puerta " + puerta + "\n");
471 } catch (Exception ex) {
472     Logger.getLogger(login.class.getName()).log(Level.SEVERE, null, ex);
473 }
474 this.txt_usuario.setText("");
475 this.txt_pass.setText("");
476 this.txt_puerta.setText("");
477 }
478 } else {
479     JOptionPane.showMessageDialog(null, "Datos incorrectos");
480 }
481
482 } catch (Exception e) {
483     JOptionPane.showMessageDialog(null, "Error al encontrar al usuario " + e.getMessage());
484 }
485

```

Ilustración 31 Método validar usuario parte 3

Con esto último se tiene todos los métodos necesarios en el apartado de la ventana login.

Diseño de ventana menu y su programación.

Ejemplificando la forma de como se administran los elementos en la ventana menú.



Ilustración 32 Ventana Menú

Y su lista de elementos es la siguiente:

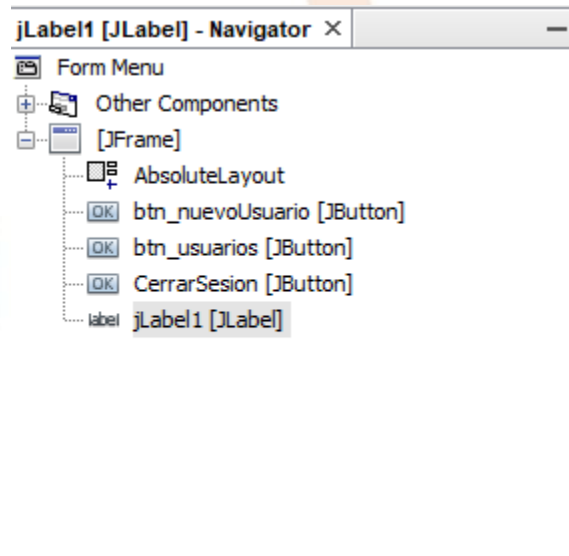


Ilustración 33 Lista de elementos de la ventana Menú

Continuando con su programación; comenzando con la configuración de las dimensiones de la ventana y anidando las imágenes correspondientes para el icono y logo de la empresa.



```

1 package controlacceso;
2
3 import javax.swing.ImageIcon;
4
5 public class Menu extends javax.swing.JFrame {
6
7     public Menu() {
8         initComponents();
9         this.setTitle("Menú Principal");
10        this.setLocation(400,200);
11        this.setResizable(false);
12        ImageIcon icono = new ImageIcon("C:\\Users\\HP\\Documents\\NetBeansProjects\\pruebaLeds\\src\\Imagenes\\agros.png");
13        this.setIconImage(icono.getImage());
14    }
15
16    @SuppressWarnings("unchecked")
17    Generated Code
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61 private void btn_nuevoUsuarioActionPerformed(java.awt.event.ActionEvent evt) {
62     //CAMBIO DE VENTANA A REGISTRO DE USUARIOS
63     this.dispose();
64     RegistrarUsuarios cu = new RegistrarUsuarios();
65     cu.setVisible(true);
66 }
67
68 private void btn_usuariosActionPerformed(java.awt.event.ActionEvent evt) {
69     // CAMBIO DE VENTANA A TABLA DE USUARIOS
70     this.dispose();
71     consultaUsuarios cu = new consultaUsuarios();
72     cu.setVisible(true);
73 }
74

```

Ilustración 34 Configuración de dimensiones de ventana Menú

Se comparte de la misma manera, la programación para las acciones correspondientes de los botones “REGISTRAR NUEVO USUARIO” y “USUARIOS REGISTRADOS”, los cuales redireccionan a las ventanas Registrar Usuarios y Consulta Usuarios respectivamente. Culminando con la acción del botón “CERRAR SESIÓN”:

```

74
75 private void CerrarSesionActionPerformed(java.awt.event.ActionEvent evt) {
76     // Cierre de sesion
77     System.exit(0);
78 }
79

```

Ilustración 35 Acción botón usuario de la ventana Menú

Diseño de ventana RegistrarUsuarios y su programación.

Iniciando con el diseño de la ventana Registrar Usuarios:

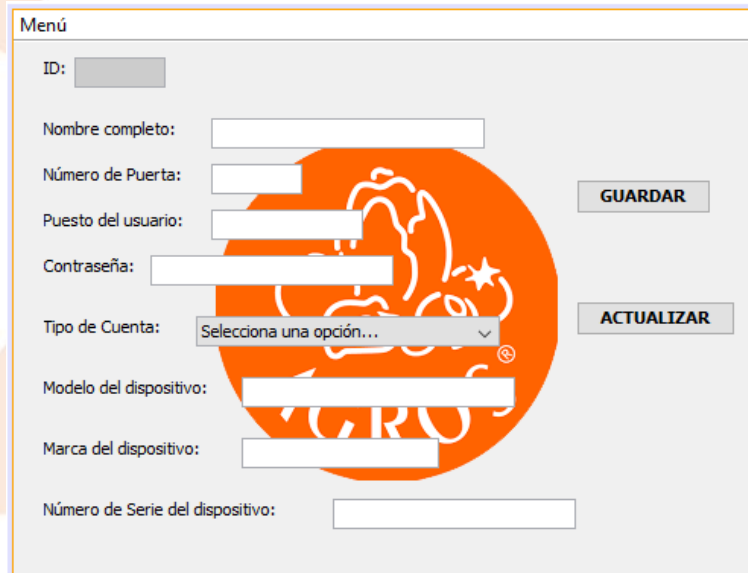


Ilustración 36 Diseño de la ventana Registro de Usuarios

Prosiguiendo con los elementos que están colocados en esta ventana:

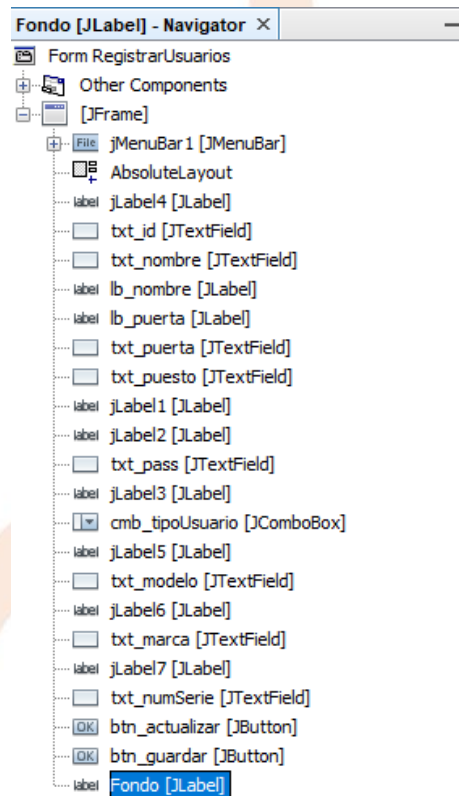


Ilustración 37 Lista de elementos de la ventana Registrar Usuarios



Ahora se analiza la programación de la ventana, comenzando con la importaciones de librerías que se usan en la ventana, continuado por la declaración de las variables universales que se estarán usando a lo largo de la programación de los métodos.

```

1 package controlacceso;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8 import java.util.logging.Level;
9 import java.util.logging.Logger;
10 import javax.swing.ImageIcon;
11 import javax.swing.JOptionPane;
12 import javax.swing.table.DefaultTableModel;
13
14 public class RegistrarUsuarios extends javax.swing.JFrame {
15
16     //Inicializamos variables y sus valores
17     String titulos[] = {"id usuario", "Nombre", "Numero de Puerta", "Puesto", "Contraseña",
18         "Tipo de Usuario", "Modelo del Dispositivo", "Marca del Dispositivo", "Número de Serie del Dispositivo"};
19     String fila[] = new String[9];
20     DefaultTableModel modelo;
21     Connection con = null;
22     Statement stmt = null;
23     String var, var2;
24
25     public RegistrarUsuarios() {
26         initComponents();
27         // Configuración de la ventana
28         this.setTitle("Registro de Usuarios");
29         this.setLocation(400, 200);
30         this.setResizable(false);
    
```

Ilustración 38 Dimensiones de la ventana Registrar usuarios

Se continua con la edición de las dimensiones de la ventana:

```

31     ImageIcon icono = new ImageIcon("C:\\Users\\HP\\Documents\\NetBeansProjects\\pruebaLeds\\src\\Imagenes\\agros.png");
32     this.setIconImage(icono.getImage());
33 }
34
35 @SuppressWarnings("unchecked")
36 // Generated Code
37
38 //Método Actualizar
39 public void actualizar() {
40     //variables de control de recuperación de datos
41     String cadena1, cadena2, cadena3, cadena4, cadena5, cadena6, cadena7, cadena8, cadena9;
42     //Inicialización de variables de control
43     cadena1 = txt_id.getText();
44     cadena2 = txt_nombre.getText();
45     cadena3 = txt_puerta.getText();
46     cadena4 = txt_puesto.getText();
47     cadena5 = txt_pass.getText();
48     cadena6 = cmb_tipoUsuario.getSelectedItem().toString();
49     cadena7 = txt_modelo.getText();
50     cadena8 = txt_marca.getText();
51     cadena9 = txt_numSerie.getText();
52
53     //Condición de validación información en los campos
54     if (txt_nombre.getText().equals("")) {
55         //ventana de instrucción
56         javax.swing.JOptionPane.showMessageDialog(this, "1.- Seleccione la opción Menú\n"
57             + "2.- Seleccione la opción Consultar Usuario Especifico\n"
58             + "3.- Ingrese el nombre completo del usuario\n"
59             + "4.- Actualice el dato deseado en el campo correspondiente",
60             "AVISO!", javax.swing.JOptionPane.INFORMATION_MESSAGE);
    
```

Ilustración 39 Método actualizar de la ventana registrar usuarios





Comenzando con el método actualizar que se encarga de actualizar los datos modificados en el registro de un usuario, una vez mandados a llamar sus datos a sus correspondientes casillas, usando otro método.

```

218         "AVISO!", javax.swing.JOptionPane.INFORMATION_MESSAGE);
219     } else {
220         //CONEXIÓN PARTICULAR A LA BASE DE DATOS
221         try {
222             String url = "jdbc:mysql://localhost:3306/controlacceso"; //NOMBRE DE LA BASE DE DATOS
223             String usuario = "root"; //NOMBRE DE USUARIO EN MYSQL
224             String contraseña = "123root"; //CONTRASEÑA DE MYSQL
225             //IMPORTACIÓN DE CLASE ENLACE DE ENVIO DE DATOS
226             Class.forName("com.mysql.jdbc.Driver").newInstance();
227             con = DriverManager.getConnection(url, usuario, contraseña);
228             if (con != null) //MENSAJE VERIFICADOR EN CONSOLA
229             {
230                 System.out.println("Se ha establecido una conexión a la base de datos "
231                     + "\n " + url);
232             }
233             stmt = con.createStatement();
234             //ACTUALIZACIÓN DE DATOS
235             stmt.executeUpdate("update ignore usuarios set id= '" + cadena1 + "', "
236                 + "nombre = '" + cadena2 + "', puerta = '" + cadena3 + "', puesto = '" + cadena4
237                 + "', pass = '" + cadena5 + "', tipousuario = '" + cadena6 + "', modelo = '" + cadena7
238                 + "', marca = '" + cadena8 + "', serie = '" + cadena9 + "'"
239                 + " where id = '" + txt_id.getText() + "' || nombre = '"
240                 + txt_nombre.getText() + "' || puerta = '" + txt_puerta.getText()
241                 + "' || puesto = '" + txt_puesto.getText() + "' || pass = '" + txt_pass.getText()
242                 + "' || tipousuario = '" + cmb_tipoUsuario.getSelectedItem() + "' || modelo = '"
243                 + txt_modelo.getText() + "' || marca = '" + txt_marca.getText() + "' || serie = '"
244                 + txt_numSerie.getText() + "'");
245             //MENSAJE ACTUALIZACIÓN DE DATOS
246             System.out.println("Los valores han sido Actualizados");
247         } catch (SQLException e) {
248             e.printStackTrace();
249         } catch (ClassNotFoundException ex) {
250             Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
251         } catch (InstantiationException ex) {
252             Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);

```

Ilustración 40 Método actualizar de la ventana menú parte 2

```

252     Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
253 } catch (IllegalAccessException ex) {
254     Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
255 } finally {
256     //CIERRE DE CONEXION CON LA BASE DE DATOS
257     if (con != null) {
258         try {
259             con.close();
260             stmt.close();
261         } catch (Exception e) {
262             System.out.println(e.getMessage());
263         }
264     }
265 }
266 //MENSAJE PARA EL USUARIO
267 javax.swing.JOptionPane.showMessageDialog(this, "Actualizado correctamente!",
268     "AVISO!", javax.swing.JOptionPane.INFORMATION_MESSAGE);
269 }
270 //LIMPIEZA DE CAMPOS
271 this.txt_id.setText("");
272 this.txt_nombre.setText("");
273 this.txt_puerta.setText("");
274 this.txt_puesto.setText("");
275 this.txt_pass.setText("");
276 this.txt_modelo.setText("");
277 this.txt_marca.setText("");
278 this.txt_numSerie.setText("");
279 }
    
```

Ilustración 41 Método actualizar datos de la ventana menú parte 3

Prosiguiendo con el método consultar que se utiliza para poder mandar a llamar los datos de un usuario, usando únicamente el nombre, verificando que esté correctamente escrito. Por medio el menú que se encuentra en la parte izquierda superior de la ventana a como se muestra a continuación:

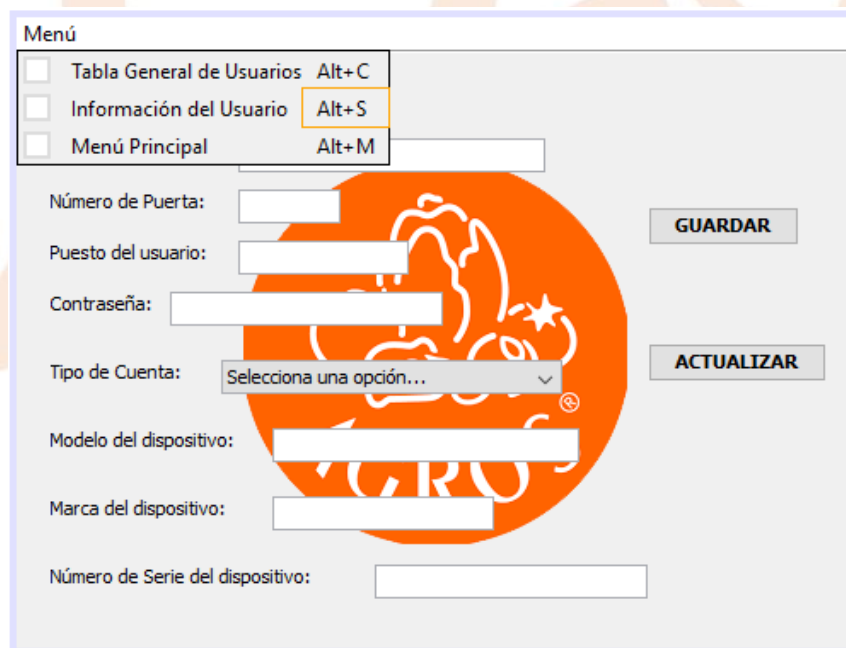


Ilustración 42 Opción de consulta de datos de un usuario


```

281 // Metodo Consultar
282 public void consulta() {
283     //DECLARACIÓN DE VARIABLES
284     String cap = "";
285     ResultSet rs = null;
286     var2 = var;
287     //CONSULTA A LA BASE DE DATOS
288     String sql2 = "Select id, nombre, puerta, puesto, pass, tipousuario, "
289         + "modelo, marca,serie FROM usuarios where nombre = '" + var2 + "'";
290
291     try {
292         //CONEXIÓN PARTICULAR CON LA BASE DE DATOS
293         String url = "jdbc:mysql://localhost:3306/controlacceso"; //NOMBRE DE LA BASE DE DATOS
294         String usuario = "root"; //USUARIO DE MYSQL
295         String contraseña = "l23root"; //CONTRASEÑA DE MYSQL
296
297         // CLASE PARA EL ENVIO DE DATOS DE LA BASE DE DATOS
298         Class.forName("com.mysql.jdbc.Driver").newInstance();
299         // VARIABLE DE CONTROL DE DATOS
300         con = DriverManager.getConnection(url, usuario, contraseña);
301         //CONDICIÓN DEL MANEJO DE DATOS
302         if (con != null) //MENSAJE DE VERIFICACIÓN EN CONSOLA
303         {
304             System.out.println("Se ha establecido una conexión a la base de datos \n " + url);
305         }
306         //VARIABLES DE MANEJO DE LA BASE DE DATOS
307         stmt = con.createStatement();
308         rs = stmt.executeQuery(sql2);
309         //VARIABLE CONTADOR
310         int i = 1;
311         //CONDICION DE BUSQUEDA
312         while (rs.next()) {
313             // VARIABLES DE CONTROL DE DATOS
314             String id = rs.getString("id");
315             String inom = rs.getString("nombre");

```

Ilustración 43 Método consultar de la ventana Registrar usuarios

Procediendo a mandar a llamar todos los datos que se encuentran registrados junto al nombre del usuario, de esta forma, haciendo posible el editar los datos registrados y actualizarlos sin complicaciones.



```

316 String ipuerta = rs.getString("puerta");
317 String ipuesto = rs.getString("puesto");
318 String ipass = rs.getString("pass");
319 String itipo = rs.getString("tipousuario");
320 String imodelo = rs.getString("modelo");
321 String imarca = rs.getString("marca");
322 String inumero_serie = rs.getString("serie");
323 System.out.println("Sitio Web " + (i++) + ":\n"
324     + id + "\n"
325     + inom + "\n"
326     + ipuerta + "\n"
327     + ipuesto + "\n"
328     + ipass + "\n"
329     + itipo + "\n"
330     + imodelo + "\n"
331     + imarca + "\n"
332     + inumero_serie + "\n\n");
333 //ENVIO DE DATOS A LOS CAMPOS
334 txt_id.setText(id);
335 txt_nombre.setText(inom);
336 txt_puerta.setText(ipuerta);
337 txt_puesto.setText(ipuesto);
338 txt_pass.setText(ipass);
339 cmb_tipoUsuario.setSelectedItem(itipo);
340 txt_modelo.setText(imodelo);
341 txt_marca.setText(imarca);
342 txt_numSerie.setText(inumero_serie);
343
344 }
345 } catch (SQLException ex) {
346     ex.printStackTrace();
347 } catch (InstantiationException ex) {
348     Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
349 } catch (IllegalAccessException ex) {

```

Ilustración 44 Método consultar de la ventana Registrar usuarios parte 2

```

350     Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
351 } catch (ClassNotFoundException ex) {
352     Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
353 } finally {
354     //DESCONEXION CON LA BASE DE DATOS
355     if (rs != null) {
356         try {
357             rs.close();
358         } catch (SQLException ex) {
359             System.out.println(ex.getMessage());
360             ex.printStackTrace();
361         }
362     }
363     if (stmt != null) {
364         try {
365             stmt.close();
366         } catch (SQLException ex) {
367             System.out.println(ex.getMessage());
368             ex.printStackTrace();
369         }
370     }
371     if (con != null) {
372         try {
373             con.close();
374         } catch (SQLException ex) {
375             System.out.println(ex.getMessage());
376             ex.printStackTrace();
377         }
378     }
379 }
380 }
    
```

Ilustración 45 Método consultar de la ventana Registrar usuarios parte 3

Continuando con poder cambiar de ventana a la de consultar usuarios por medio de una opción del menú y accionándola:

```

382 private void consultaGeneralActionPerformed(java.awt.event.ActionEvent evt) {
383     //CAMBIO DE VENTANA A TABLA GENERAL DE USUARIOS
384     this.dispose();
385     consultaUsuarios cu = new consultaUsuarios();
386     cu.setVisible(true);
387 }
    
```

Ilustración 46 Cambio de ventana Registrar usuarios a Consultar usuarios

Siguiendo el método para mandar a llamar datos de un usuario específico:

```

389 private void consultarUsuarioActionPerformed(java.awt.event.ActionEvent evt) {
390     // OPCIÓN DE MENÚ Consultar INFORMACIÓN DE USUARIO
391     //DECLARACIÓN DE VARIABLES
392     String cap = "";
393     ResultSet rs = null;
394     //VENTANA DE REQUISICIÓN DE DATOS
395     var = javax.swing.JOptionPane.showInputDialog(this, "Nombre del usuario", "Consulta usuario",
396         javax.swing.JOptionPane.QUESTION_MESSAGE);
397     //VARIABLE DE CONSULTA DE LA BASE DE DATOS
398     String sql = "SELECT* FROM usuarios WHERE nombre = '" + var + "'";
399     //CONDICION DE CANCELACIÓN
400     if (var == null) {
401         javax.swing.JOptionPane.showMessageDialog(this, "La accion fue cancelada", "AVISO!",
402             javax.swing.JOptionPane.INFORMATION_MESSAGE);
403     } else {
404         //CONDICION DE CAMPO
405         if (var.equals("")) {
406             javax.swing.JOptionPane.showMessageDialog(this,
407                 "Favor de ingresar el nombre de usuario\nque desea consultar", "AVISO!",
408                 javax.swing.JOptionPane.INFORMATION_MESSAGE);
409         } else {
410             try {
411                 //CONEXION PARTICULAR CON LA BASE DE DATOS
412                 String url = "jdbc:mysql://localhost:3306/controlacceso";
413                 String usuario = "root";
414                 String contraseña = "123root";
415                 //CLASE PARA EL ENVIO DE DATOS DE LA BD
416                 Class.forName("com.mysql.jdbc.Driver").newInstance();
417                 //VARIABLE DE CONEXION DE DATOS
418                 con = DriverManager.getConnection(url, usuario, contraseña);
419                 if (con != null) {
420                     System.out.println("Se ha establecido una conexión a la base de datos "
421                         + "\n" + url);
422                 }

```

Ilustración 47 Método consultar la información de un usuario

```

423         stmt = con.createStatement();
424         rs = stmt.executeQuery(sql);
425
426         while (rs.next()) {
427             //VARIABLE DE VALIDACIÓN DE TIPO DE USUARIO
428             cap = rs.getString("tipousuario");
429             //CONDICIÓN DE TIPO DE USUARIO
430             if (cap.equals("Usuario") || cap.equals("Administrador")) {
431                 consulta();
432             }
433         } // fin del bucle While
434
435     } catch (InstantiationException | IllegalAccessException | ClassNotFoundException | SQLException ex) {
436         Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
437     } finally {
438         //DESCONEXIÓN CON LA BASE DE DATOS
439         if (con != null) {
440             try {
441                 con.close();
442                 stmt.close();
443             } catch (Exception e) {
444                 System.out.println(e.getMessage());
445             }
446         }
447     }
448
449     //CONDICIÓN SI EL USUARIO NO EXISTE
450     if (!cap.equals("Usuario") && !cap.equals("Administrador")) {
451         //MENSAJE DE ERROR PARA EL USUARIO
452         javax.swing.JOptionPane.showMessageDialog(this, "El usuario no fue encontrado\n",
453             "ERROR!", javax.swing.JOptionPane.ERROR_MESSAGE);
454     }
455 }
456
457 }

```

Ilustración 48 Método consultar la información de un usuario

Prosiguiendo con la configuración de algunas opciones como cambiar a la ventana de menú y los tipos de caracteres aceptados en las casillas de texto.

```

private void MenuPrincipalActionPerformed(java.awt.event.ActionEvent evt) {
    // CAMBIO DE VENTANA A MENU PRINCIPAL DEL ADMINISTRADOR
    this.dispose();
    Menu cu = new Menu();
    cu.setVisible(true);
}

private void consultaDeUsuariosActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void txt_nombreActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void txt_nombreKeyTyped(java.awt.event.KeyEvent evt) {
    //ACEPTACION DE SOLO CARACTERES ALFABÉTICOS Y ESPACIOS EN BLANCO
    char c = evt.getKeyChar();
    if ((c < 'a' || c > 'z') && (c < 'A' || c > 'Z') && (c < ' ' || c > ' ')) {
        evt.consume();
    }
}

private void txt_puertaKeyTyped(java.awt.event.KeyEvent evt) {
    //ACEPTACION DE SOLO CARACTERES NUMÉRICOS
    char c = evt.getKeyChar();
    if ((c < '0' || c > '9')) {
        evt.consume();
    }
}
    
```

Ilustración 49 Cambio a ventana menú y tipos de caracteres aceptados

```

private void txt_passKeyTyped(java.awt.event.KeyEvent evt) {
    //ACEPTACION DE SOLO CARACTERES NUMÉRICOS
    char c = evt.getKeyChar();
    if ((c < '0' || c > '9')) {
        evt.consume();
    }
}

private void btn_actualizarActionPerformed(java.awt.event.ActionEvent evt) {
    //METODO ACTUALIZAR
    actualizar();
}
    
```

Ilustración 50 Tipos de caracteres aceptados y ejecución del método actualizar

Por último, tenemos la acción del botón de guardado de información para el caso de un usuario nuevo:

```
503 private void btn_guardarActionPerformed(java.awt.event.ActionEvent evt) {
504     // DECLARACION DE VARIABLES
505     String cadena1, cadena2, cadena3, cadena4, cadena5, cadena6, cadena7, cadena8;
506     //VALORES DE LAS VARIABLES
507     cadena1 = txt_nombre.getText();
508     cadena2 = txt_puerta.getText();
509     cadena3 = txt_puesto.getText();
510     cadena4 = txt_pass.getText().toString();
511     cadena5 = cmb_tipoUsuario.getSelectedItem().toString();
512     cadena6 = txt_modelo.getText();
513     cadena7 = txt_marca.getText();
514     cadena8 = txt_numSerie.getText();
515     // CONDICION DE LOS CAMPOS EN BLANCO
516     if (txt_nombre.getText().equals("") || (txt_puerta.getText().equals("")) || (txt_puesto.getText().equals(""))
517         || (txt_pass.getText().equals(""))
518         || (cmb_tipoUsuario.getSelectedItem().equals("Selecciona una opción..."))
519         || (txt_modelo.getText().equals("")) || (txt_marca.getText().equals(""))
520         || (txt_numSerie.getText().equals(""))) {
521         javax.swing.JOptionPane.showMessageDialog(this, "Debe llenar todos los campos \n",
522             "ADVERTENCIA", javax.swing.JOptionPane.INFORMATION_MESSAGE);
523         txt_nombre.requestFocus();
524     }else{
525         try {
526             //CONEXION PARTICULAR CON LA BASE DE DATOS
527             String url = "jdbc:mysql://localhost:3306/controlacceso";//NOMBRE DE LA BASE DE DATOS
528             String usuario = "root"; //USUARIO DE MYSQL
529             String contraseña = "123root"; //CONTRASEÑA DE MYSQL
530             //CLASE DE ENVIO DE DATOS DE LA BASE DE DATOS
531             Class.forName("com.mysql.jdbc.Driver").newInstance();
532             //VARIABLE DE CONEXIÓN CON LA BASE DE DATOS
533             con = DriverManager.getConnection(url,usuario,contraseña);
534
535             //CONDICIÓN DE CONEXIÓN CON LA BD
536             if (con != null) System.out.println("Se ha establecido una conexión a la base de datos " +
537                 "\n " + url );
```

Ilustración 51 Guardar datos de un nuevo usuario

```
538         stmt = con.createStatement();
539         stmt.executeUpdate("INSERT INTO usuarios VALUES('" + 0 + "','" +cadena1+"','" +cadena2+"','"
540             +cadena3+"','" +cadena4+"','" +cadena5+"','" +cadena6+"','" +cadena7+"','" +cadena8+"')");
541         //MENSAJE DE VALIDACIÓN DE DATOS DE LA BD
542         System.out.println("Los valores han sido agregados a la base de datos ");
543
544     } catch (InstantiationException ex) {
545         Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
546     } catch (IllegalAccessException ex) {
547         Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
548     } catch (ClassNotFoundException ex) {
549         Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
550     } catch (SQLException ex) {
551         Logger.getLogger(RegistrarUsuarios.class.getName()).log(Level.SEVERE, null, ex);
552     }
553     finally {
554         //CONDICION DE ERROR
555         if (con != null) {
556             try {
557                 con.close();
558                 stmt.close();
559             } catch (Exception e) {
560                 System.out.println(e.getMessage());
561             }
562         }
563     }
564     //MENSAJE EXITOSO PARA EL USUARIO
565     javax.swing.JOptionPane.showMessageDialog(this,"Registro exitoso! \n",
566         "AVISO!",javax.swing.JOptionPane.INFORMATION_MESSAGE);
567
568     //LIMPIEZA DE CAMPOS
569     this.txt_nombre.setText("");
570     this.txt_puerta.setText("");
571     this.txt_puesto.setText("");
572     this.txt_pass.setText("");
```

Ilustración 52 Guardar datos de un nuevo usuario parte 2

Diseño de ventana ConsultaUsuarios y su programación.

Se muestra la forma gráfica de ubicación de elementos gráficos en la ventana:

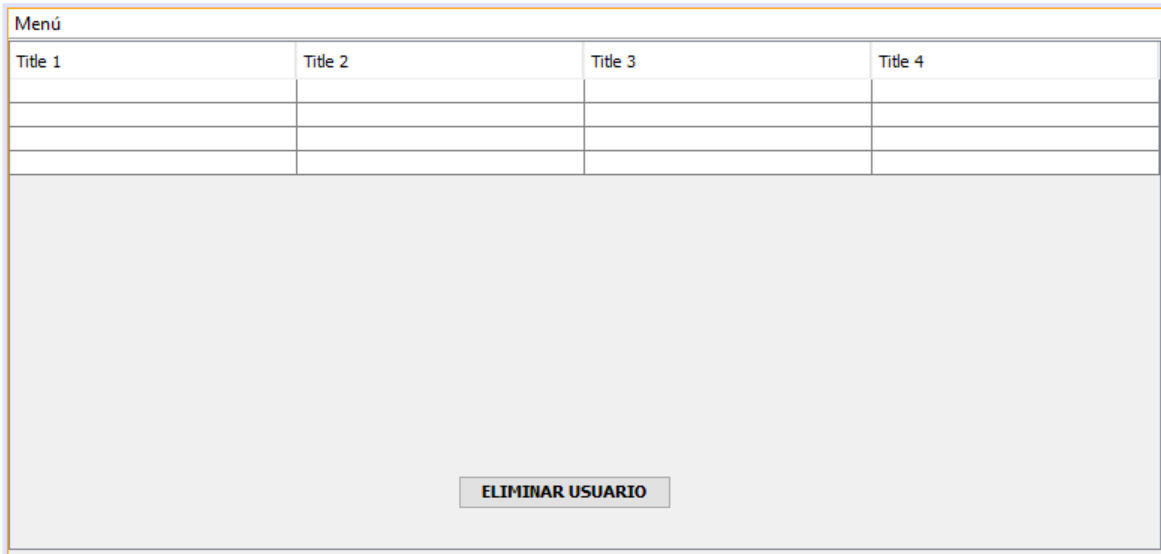


Ilustración 53 Diseño gráfico de la ventana de consulta de usuarios

Lista de elementos que se tienen añadidos en la ventana consultar usuarios:

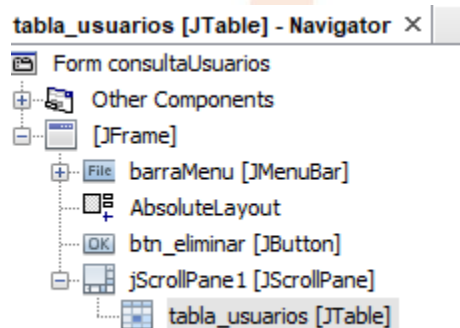


Ilustración 54 Elementos de la ventana consultar usuarios

Para la programación de esta ventana se comienza con la declaración de variables universales usadas a lo largo de la ventana, las dimensiones de la ventana y sus iconos; por último, haciendo una conexión directa con la base de datos.



```

consultaUsuarios.java
Source Design History
1 package controlacceso;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.Statement;
7 import javax.swing.ImageIcon;
8 import javax.swing.JOptionPane;
9 import javax.swing.table.DefaultTableModel;
10 import javax.swing.table.TableColumn;
11
12
13 public class consultaUsuarios extends javax.swing.JFrame {
14     //VARIABLES UNIVERSALES
15     Connection con = null;
16     Statement stmt = null;
17     String titulos[] = {"id", "Nombre", "Puerta", "Puesto", "Contraseña", "Tipo usuario",
18         "Modelo del Dispositivo", "Marca del Dispositivo", "Número de Serie del Dispositivo"};
19     String fila[] = new String (9);
20     DefaultTableModel modelo;
21     String var, var2;
22
23     public consultaUsuarios() {
24         initComponents();
25         //CONFIGURACIONES DE LA VENTANA
26         this.setTitle("Consultar Usuarios");
27         this.setLocation(300,200);
28         this.setResizable(false);
29         ImageIcon icono = new ImageIcon("C:\\Users\\HP\\Documents\\NetBeansProjects\\pruebaLeds\\src\\Imagenes\\agros.png");
30         this.setIconImage(icono.getImage());
31
32         try {
33             //CONEXIÓN CON LA BASE DE DATOS
34             String url = "jdbc:mysql://localhost:3306/controlacceso";//NOMBRE DE LA BASE DE DATOS
35             String usuario = "root";//USUARIO DE MYSQL

```

Ilustración 55 Programación de ventana consultar usuarios

Siguiendo con una sentencia SQL que nos proporcione todos los datos de todos los usuarios registrados dentro de la base de datos.

```

35 String usuario = "root";//USUARIO DE MYSQL
36 String contraseña = "123root";//CONTRASEÑA DE MYSQL
37 //CLASE PARA EL ENVÍO DE DATOS
38 Class.forName("com.mysql.jdbc.Driver").newInstance();
39 con = DriverManager.getConnection(url,usuario,contraseña);
40 if (con!= null)
41     //MENSAJE DE VALIDACIÓN EN CONSOLA
42     System.out.println("Se ha establecido una conexion a la base de datos"+ "\n"+url);
43
44     stmt = con.createStatement();
45     //CONSULTA PARA LA BASE DE DATOS
46     ResultSet rs = stmt.executeQuery("select* from usuarios");
47     //VARIABLE DE LOS TITULOS DE LA TABLA
48     modelo = new DefaultTableModel(null,titulos);
49     //CICLO DE BÚSQUEDA DE DATOS
50     while(rs.next()) {
51
52         fila[0] = rs.getString("id");
53         fila[1] = rs.getString("nombre");
54         fila[2] = rs.getString("puerta");
55         fila[3] = rs.getString("puesto");
56         fila[4] = rs.getString("pass");
57         fila[5] = rs.getString("tipousuario");
58         fila[6] = rs.getString("modelo");
59         fila[7] = rs.getString("marca");
60         fila[8] = rs.getString("serie");
61         //CREACION DE LAS FILAS DE LA TABLA
62         modelo.addRow(fila);
63     }
64     tabla_usuarios.setModel(modelo);
65     //TITULOS DE LA TABLA
66     TableColumn ci = tabla_usuarios.getColumn("id");
67     ci.setMaxWidth(25);
68     TableColumn cn = tabla_usuarios.getColumn("Nombre");
69     cn.setMaxWidth(170);

```

Ilustración 56 Llamar los datos de todos los usuarios en la base de datos

Y se mandan hacia la tabla de la ventana para poder ser mostrados de manera fácil y rápida.

```

70      TableColumn cd = tabla_usuarios.getColumn("Puerta");
71      cd.setMaxWidth(50);
72      TableColumn ct = tabla_usuarios.getColumn("Puesto");
73      ct.setMaxWidth(150);
74      TableColumn cnick = tabla_usuarios.getColumn("Contraseña");
75      cnick.setMaxWidth(90);
76      TableColumn ctipo = tabla_usuarios.getColumn("Tipo usuario");
77      ctipo.setMaxWidth(95);
78      TableColumn cmodelo = tabla_usuarios.getColumn("Modelo del Dispositivo");
79      cmodelo.setMaxWidth(500);
80      TableColumn cmarca = tabla_usuarios.getColumn("Marca del Dispositivo");
81      cmarca.setMaxWidth(500);
82      TableColumn cnumSerie = tabla_usuarios.getColumn("Número de Serie del Dispositivo");
83      cnumSerie.setMaxWidth(500);
84
85  }
86  catch (Exception e) {
87      //MENSAJE DE ERROR CUANDO NO SE PUEDEN RECUPERAR LOS DATOS DE LA BD
88      JOptionPane.showMessageDialog(null, "Error al extraer los datos de la tabla");
89  }
90
91  }
    
```

Ilustración 57 Mandar datos a la tabla

Prosigue el método de eliminación de usuarios:

```

159  //METODO ELIMINAR REGISTRO
160  public void eliminar(){
161      try {
162          //VARIALE DE SELECCIÓN EN LA TABLA
163          int filaSeleccionada = tabla_usuarios.getSelectedRow();
164          //CONSULTA PARA LA BASE DE DATOS
165          String sql="DELETE FROM usuarios WHERE id="+tabla_usuarios.getValueAt(filaSeleccionada, 0);
166          Statement st= con.createStatement();
167          int n=st.executeUpdate(sql);
168          //CONDICION DE ELIMINACIÓN DE REGISTRO
169          if (n>=0) {
170              //MENSAJE DE CONFIRMACIÓN PARA EL USUARIO
171              JOptionPane.showMessageDialog(null, "Usuario eliminado satisfactoriamente");
172          }
173      } catch (Exception e) {
174          //MENSAJE DE ERROR AL ELIMINAR EL REGISTRO
175          JOptionPane.showMessageDialog(null, "Error al eliminar al Usuario");
176      }
177  }
    
```

Ilustración 58 Método eliminar usuario

Y el último método generado es actualizar, que se encarga volver a cargar los datos a la tabla para poder observar el cambio en los registros, después de haber eliminado a algún usuario.



```

179 //METODO ACTUALIZAR
180 public void actualizar() {
181     try {
182         //CONEXIÓN CON LA BASE DE DATOS
183         String url = "jdbc:mysql://localhost:3306/controlacceso";//NOMBRE DE LA BASE DE DATOS
184         String usuario = "root"; //USUARIO DE MYSQL
185         String contraseña = "123root"; //CONTRASEÑA DE MYSQL
186         //CLASE DE ENVIO DE DATOS
187         Class.forName("com.mysql.jdbc.Driver").newInstance();
188         con = DriverManager.getConnection(url,usuario,contraseña);
189         if (con!= null)
190             //MENSAJE DE VALIDACIÓN CON LA BASE DE DATOS
191             System.out.println("Se ha establecido una conexion a la base de datos"+ "\n"+url);
192         stmt = con.createStatement();
193         //CONSULTA PARA LA BD
194         ResultSet rs = stmt.executeQuery("select* from usuarios");
195         //CREACION DE TITULOS DE LA BD
196         modelo = new DefaultTableModel(null,titulos);
197         //CICLO DE BUSQUEDA DE DATOS
198         while(rs.next()) {
199             //GUARDAR DATOS DE DE FILAS DE LA BD
200             fila[0] = rs.getString("id");
201             fila[1] = rs.getString("nombre");
202             fila[2] = rs.getString("puerta");
203             fila[3] = rs.getString("puesto");
204             fila[4] = rs.getString("pass");
205             fila[5] = rs.getString("tipousuario");
206             fila[6] = rs.getString("modelo");
207             fila[7] = rs.getString("marca");
208             fila[8] = rs.getString("serie");
209             //CREACION DE FILAS DE LA TABLA
210             modelo.addRow(fila);
211         }
212         //ESPECIFICACIONES DE LA FILAS DE LA TABLA
213         tabla usuarios.setModel(modelo);

```

Ilustración 59 Método actualizar

```

214 TableColumn ci = tabla_usuarios.getColumn("id");
215 ci.setMaxWidth(25);
216 TableColumn cn = tabla_usuarios.getColumn("Nombre");
217 cn.setMaxWidth(170);
218 TableColumn cd = tabla_usuarios.getColumn("Puerta");
219 cd.setMaxWidth(50);
220 TableColumn ct = tabla_usuarios.getColumn("Puesto");
221 ct.setMaxWidth(150);
222 TableColumn cnick = tabla_usuarios.getColumn("Contraseña");
223 cnick.setMaxWidth(90);
224 TableColumn ctipo = tabla_usuarios.getColumn("Tipo usuario");
225 ctipo.setMaxWidth(95);
226 TableColumn cmodelo = tabla_usuarios.getColumn("Modelo del Dispositivo");
227 cmodelo.setMaxWidth(1600);
228 TableColumn cmarca = tabla_usuarios.getColumn("Marca del Dispositivo");
229 cmarca.setMaxWidth(1600);
230 TableColumn cnumSerie = tabla_usuarios.getColumn("Número de Serie del Dispositivo");
231 cnumSerie.setMaxWidth(1600);
232
233 }
234 catch (Exception e) {
235     //MENSAJE DE ERROR AL OBTENER LOS DATOS DE LA BD
236     JOptionPane.showMessageDialog(null,"Error al extraer los datos de la tabla");
237 }
238 }

```

Ilustración 60 Método actualizar parte 2

Siendo lo ultimo las configuraciones para los cambios de ventana a la ventana registro de usuarios y menú, de la misma forma que, la acción al presionar el botón eliminar:

```

239
240 [-] private void menuRegistroActionPerformed(java.awt.event.ActionEvent evt) {
241     // CAMBIO DE VENTANA A REGISTRO DE USUARIOS
242     this.dispose();
243     RegistrarUsuarios u = new RegistrarUsuarios();
244     u.setVisible(true);
245 }
246
247 [-] private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
248     // CAMBIO DE VENTANA A MENÚ PRINCIPAL DEL ADMINISTRADOR
249     this.dispose();
250     Menu cu = new Menu();
251     cu.setVisible(true);
252 }
253
254 [-] private void btn_eliminarActionPerformed(java.awt.event.ActionEvent evt) {
255     //LLAMAR A LOS METODOS DE ELIMINAR Y ACTUALIZAR
256     eliminar();
257     actualizar();
258 }
    
```

Ilustración 61 Cambio a ventanas Menú, Registro de usuarios y acción al botón eliminar

Programación de la clase ConexiónDB.

Esta clase es necesaria para poder hacer más rápidos procesos en las diferentes ventanas que ya se tiene configuradas, de esta manera, solo se manda a llamar un método entre clases, de tal forma que ahorra también líneas de código en la programación de la lógica de las ventanas requeridas para el administrador.

```

1 package controlacceso;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import javax.swing.JOptionPane;
6
7 public class ConexionDB {
8     Connection conectar = null;
9     public Connection conexion() {
10         try {
11             Class.forName("com.mysql.jdbc.Driver");
12             String url = "jdbc:mysql://localhost:3306/controlacceso";
13             String usuario = "root";
14             String contraseña = "123root";
15
16             //VARIABLE DE CONEXION DE DATOS
17             conectar = (Connection) DriverManager.getConnection(url, usuario, contraseña);
18         } catch (Exception e) {
19             JOptionPane.showMessageDialog(null, "Error de conexión con la base de datos" + e.getMessage());
20         }
21         return conectar;
22     }
23 }
24
25
    
```

Ilustración 62 Clase de conexión con la base de datos

PROGRAMACIÓN EN ARDUINO

Diseño preliminar del circuito

Se muestra el diseño que se de seguir en la instalación de los componentes electrónicos en la placa Arduino, en ese caso siguiendo el ejemplo de usar una placa Arduino MEGA:

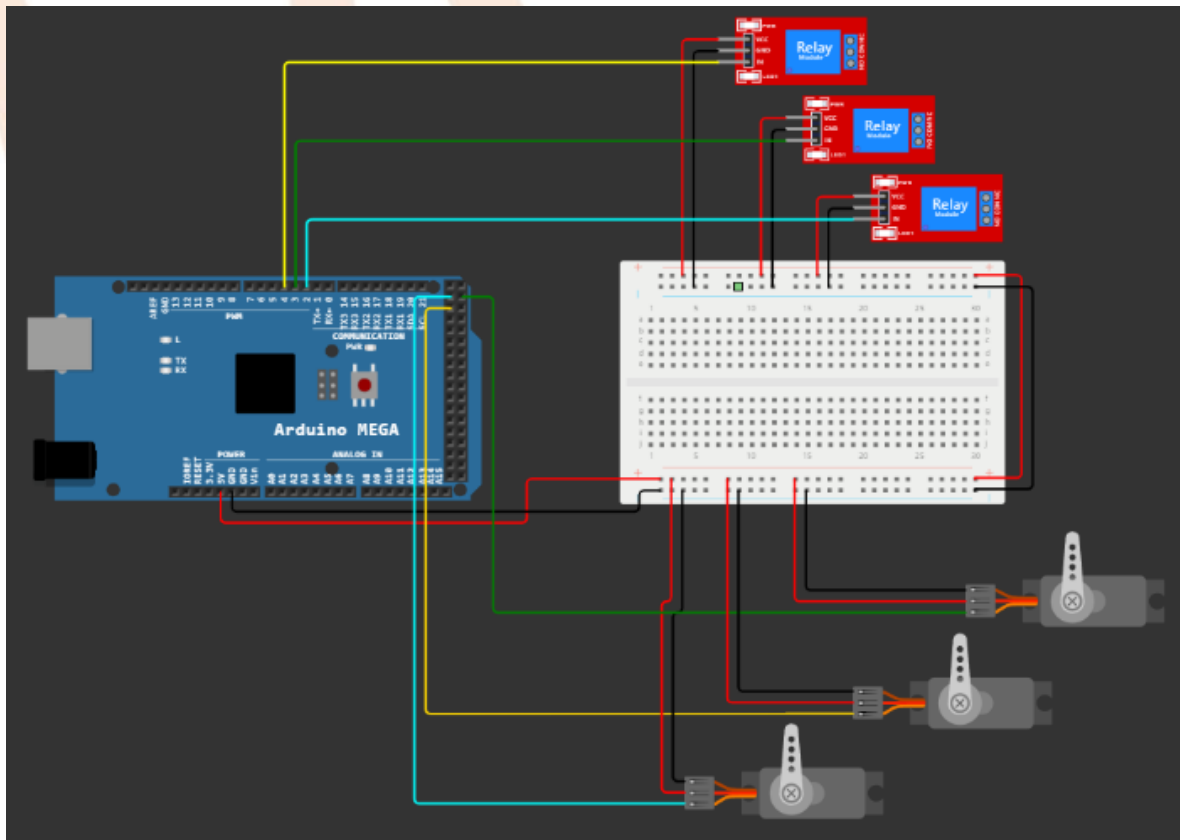


Ilustración 63 Diseño preliminar del circuito electrónico

Donde el color azul son los asignados a la puerta 1, los de color verde son para la puerta 2 y, por último, los amarillos son para la puerta 3; de esta manera, se sigue la misma lógica de configuración para los componentes totales que se necesiten según el número de puertas que tiene el diseño.

Programación en Arduino IDE

Teniendo el caso de tener 10 usuarios para asignar una puerta a cada uno, se tiene el siguiente código para administrar dichas puertas con ayuda de un Arduino MEGA; teniendo la inicialización de los 10 relevadores, los 10 servomotores y cada uno de ellos con su respectivo pin de control:

```

AccesoApuestas.ino
1  #include <Servo.h>
2
3  //INICIALIZAMOS LAS VARIABLES DE CONTROL
4  int Rele1 = 2, Rele2 = 3, Rele3 = 4, Rele4 = 5, Rele5 = 6;
5  int Rele6 = 7, Rele7 = 8, Rele8 = 9, Rele9 = 10, Rele10 = 11;
6  int servo1 = 22, servo2 = 23, servo3 = 24, servo4 = 25, servo5 = 26;
7  int servo6 = 27, servo7 = 28, servo8 = 29, servo9 = 30, servo10 = 31;
8  char mensaje;
9  Servo servomotor1, servomotor2, servomotor3, servomotor4, servomotor5;
10 Servo servomotor6, servomotor7, servomotor8, servomotor9, servomotor10;
11
12 void setup() {
13     // put your setup code here, to run once:
14     Serial.begin(9600);
15     pinMode(Rele1, OUTPUT);
16     pinMode(Rele2, OUTPUT);
17     pinMode(Rele3, OUTPUT);
18     pinMode(Rele4, OUTPUT);
19     pinMode(Rele5, OUTPUT);
20     pinMode(Rele6, OUTPUT);
21     pinMode(Rele7, OUTPUT);
22     pinMode(Rele8, OUTPUT);
23     pinMode(Rele9, OUTPUT);
24     pinMode(Rele10, OUTPUT);
25     servomotor1.attach(servo1);
26     servomotor2.attach(servo2);
27     servomotor3.attach(servo3);
28     servomotor4.attach(servo4);
29     servomotor5.attach(servo5);

```

Ilustración 64 Programación de la placa Arduino

Continuando con el ciclo de acción de los componentes eléctricos, según sea el mensaje recibido en la placa Arduino, por parte del programa de control de acceso:

```

30  servomotor6.attach(servo6);
31  servomotor7.attach(servo7);
32  servomotor8.attach(servo8);
33  servomotor9.attach(servo9);
34  servomotor10.attach(servo10);
35  }
36
37  void loop() {
38    // CONDICION PARA VERIFIACAR SI HAY UN MENSAJE ENVIADO AL ARDUINO
39    if (Serial.available() > 0) {
40      //GUARDAMOS EL MENSAJE ENVIADO
41      mensaje = Serial.read();
42      //CONDICIÓN SEGÚN EL VALOR DEL MENSAJE
43      if (mensaje == '1') {
44        digitalWrite(Rele1, HIGH);
45        servomotor1.write(1);
46        delay(10000); //tiempo de espera 10seg
47        digitalWrite(Rele1, LOW);
48        servomotor1.write(90);
49
50      } else if (mensaje == '2') {
51        digitalWrite(Rele2, HIGH);
52        servomotor2.write(1);
53        delay(10000);
54        digitalWrite(Rele2, LOW);
55        servomotor2.write(90);
56
57      } else if (mensaje == '3') {
58        digitalWrite(Rele3, HIGH);

```

Ilustración 65 Ciclo loop para el accionamiento de puertas

Siguiendo la misma sentencia hasta llegar al caso de la puerta numero 10:



```

59     servomotor3.write(1);
60     delay(10000);
61     digitalWrite(Rele3, LOW);
62     servomotor3.write(90);
63
64     }else if(mensaje == '4') {
65         digitalWrite(Rele4, HIGH);
66         servomotor4.write(1);
67         delay(10000);
68         digitalWrite(Rele4, LOW);
69         servomotor4.write(90);
70
71     }else if(mensaje == '5') {
72         digitalWrite(Rele5, HIGH);
73         servomotor5.write(1);
74         delay(10000);
75         digitalWrite(Rele5, LOW);
76         servomotor5.write(90);
77
78     }else if(mensaje == '6') {
79         digitalWrite(Rele6, HIGH);
80         servomotor6.write(1);
81         delay(10000);
82         digitalWrite(Rele6, LOW);
83         servomotor6.write(90);
84
85     }else if(mensaje == '7') {
86         digitalWrite(Rele7, HIGH);
87         servomotor7.write(1);
    
```

Ilustración 66 Accionamiento hasta la puerta 7



```

87     servomotor7.write(1);
88     delay(10000);
89     digitalWrite(Rele7, LOW);
90     servomotor7.write(90);
91
92     }else if(mensaje == '8') {
93         digitalWrite(Rele8, HIGH);
94         servomotor8.write(1);
95         delay(10000);
96         digitalWrite(Rele8, LOW);
97         servomotor8.write(90);
98
99     }else if(mensaje == '9') {
100         digitalWrite(Rele9, HIGH);
101         servomotor9.write(1);
102         delay(10000);
103         digitalWrite(Rele9, LOW);
104         servomotor9.write(90);
105
106     }else if(mensaje == '10') {
107         digitalWrite(Rele10, HIGH);
108         servomotor10.write(1);
109         delay(10000);
110         digitalWrite(Rele10, LOW);
111         servomotor10.write(90);
112     }
113 }
114

```

Ilustración 67 Accionamiento hasta la puerta 10