

**SISTEMA DE VENTA DE VIDEOJUEGOS**

**MANUAL DETALLADO**

**ELABORADO POR:**

**Luis David Alejandro Soto Lara**

**Brayan Antonio Cimentales Vidal**

**Martin Iván García Alonso**

**Antonio Rodríguez Brewster**

**SAN NICOLÁS DE LAS GARZAS, CD. UNIVERSITARIA**

**LUNES 25 DE MAYO DE 2020**

INDICE

[**1.** **INTODUCCION** 1](#_Toc41933649)

[**a.** **Objetivos específicos** 1](#_Toc41933650)

[**2.** **MODULOS** 2](#_Toc41933651)

[**2.1** **Login** 2](#_Toc41933652)

[**2.2** **Ventana Principal** 3](#_Toc41933653)

[**a.** **Submenú “Clientes”** 3](#_Toc41933654)

[**b.** **Submenú “generar venta”** 3](#_Toc41933655)

[**c.** **Submenú “Detalle ventas”** 4](#_Toc41933656)

[**d.** **Submenú “Generar factura”** 4](#_Toc41933657)

[**e.** **Submenú “Reporte Factura”** 5](#_Toc41933658)

[**f.** **Submenú “Nuevo vendedor”** 5](#_Toc41933659)

[**g.** **Submenú “Generar reporte de venta”** 6](#_Toc41933660)

[**h.** **Submenú “Editar Stock”** 6](#_Toc41933661)

[**i.** **Submenú “Agregar a Stock”** 7](#_Toc41933662)

[**2.3** **Clientes** 7](#_Toc41933663)

[**2.4** **Generar venta** 8](#_Toc41933664)

[**2.5** **Generar factura** 8](#_Toc41933665)

[**2.6** **Nuevo vendedor** 10](#_Toc41933666)

[**2.7** **Reporte ventas** 10](#_Toc41933667)

[**2.8** **Stock** 11](#_Toc41933668)

[**2.9** **Editar Stock** 12](#_Toc41933669)

1. **INTODUCCION**

Este manual le permitirá conocer las funciones principales del Sistema de Venta de Videojuegos (SDVV).

* 1. **Objetivos específicos**

Registrar las ventas realizadas durante cierto tiempo para poder generar las metas de ventas.

Poder identificar los distintos videojuegos más vendidos para poder generar más ventas.

Definir los estados de ventas y facturas.

1. **MODULOS**
   1. **Login**

CODIGO DEL BOTON “ENTRAR”

**Esta parte del código nos permite ingresar datos de usuario y contraseñas para poder entrar al sistema de ventas, en caso de no tener un usuario previamente creado no nos dejara acceder.**

Connection con=null;

PreparedStatement pst=null;

ResultSet rs=null;

try{

Class.forName("net.ucanaccess.jdbc.UcanaccessDriver");

con=DriverManager.getConnection("jdbc:ucanaccess://C:\\Users\\ALMA LARA\\Documents\\NetBeansProjects\\PROYECTO POO\\Base de datos\\BD\_Videojuegos.accdb");

String sql="select \* from login where Usuario='"+user.getText()+"' and Contraseña='"+contraseña.getText()+"'";

pst=con.prepareStatement(sql);

rs=pst.executeQuery();

if (rs.next()){

}else{

JOptionPane.showMessageDialog(this, "¡USUARIO O CONTRASEÑA INCORRECTOS!", "Error", JOptionPane.ERROR\_MESSAGE);

}

}catch(HeadlessException | ClassNotFoundException | SQLException e){

JOptionPane.showMessageDialog(null,e);

}

* 1. **Ventana Principal**
  2. **Submenú “Clientes”**

CODIGO DEL BOTON “CLIENTES”

**Esta parte del código nos permite acceder al submenú “clientes” en el cual podemos agregar a un nuevo cliente a la base de datos.**

Clientes obj = new Clientes();

VentanaPrincipal.add(obj);

obj.toFront();

obj.setLocation(VentanaPrincipal.getWidth() / 2 - obj.getWidth() / 2, VentanaPrincipal.getHeight() / 2 - obj.getHeight() / 2);

try {

obj.setMaximum(false);

} catch (PropertyVetoException ex) {

Logger.getLogger(Interfaz.class.getName()).log(Level.SEVERE, null, ex);

}

obj.setVisible(true);

* 1. **Submenú “generar venta”**

CODIGO DEL BOTON “GENERAR VENTA”

**Esta parte del código nos permite acceder al submenú “Generar venta” el cual nos permite generar y guardar una venta en la base de datos para poder ser guardada en los reportes.**

GenerarVenta obj = new GenerarVenta();

VentanaPrincipal.add(obj);

obj.toFront();

obj.setLocation(VentanaPrincipal.getWidth() / 2 - obj.getWidth() / 2, VentanaPrincipal.getHeight() / 2 - obj.getHeight() / 2);

try {

obj.setMaximum(false);

} catch (PropertyVetoException ex) {

Logger.getLogger(Interfaz.class.getName()).log(Level.SEVERE, null, ex);

}

obj.setVisible(true);

* 1. **Submenú “Detalle ventas”**

CODIGO DEL BOTON “DETALLE VENTAS”

**Esta parte del código nos permite entrar al submenú detalle ventas para poder ver las ventas generadas durante todo el año.**

detalleventas obj = new detalleventas();

VentanaPrincipal.add(obj);

obj.toFront();

obj.setLocation(VentanaPrincipal.getWidth() / 2 - obj.getWidth() / 2, VentanaPrincipal.getHeight() / 2 - obj.getHeight() / 2);

try {

obj.setMaximum(false);

} catch (PropertyVetoException ex) {

Logger.getLogger(Interfaz.class.getName()).log(Level.SEVERE, null, ex);

}

obj.setVisible(true);

* 1. **Submenú “Generar factura”**

CODIGO DEL BOTON “GENERAR VENTAS”

**Esta parte del código nos permite acceder al submenú Generar factura, para así poder crear y guardar una factura si el cliente la socita.**

detalleventas obj = new detalleventas();

VentanaPrincipal.add(obj);

obj.toFront();

obj.setLocation(VentanaPrincipal.getWidth() / 2 - obj.getWidth() / 2, VentanaPrincipal.getHeight() / 2 - obj.getHeight() / 2);

try {

obj.setMaximum(false);

} catch (PropertyVetoException ex) {

Logger.getLogger(Interfaz.class.getName()).log(Level.SEVERE, null, ex);

}

obj.setVisible(true);

* 1. **Submenú “Reporte Factura”**

CODIGO DEL BOTON “REPORTE FACTURA”

**Esta parte del código nos permite acceder al submenú reporte factura, en el cual podemos generar algún reporte de facturas según el mes que seseemos conocer.**

gfactura obj = new gfactura();

VentanaPrincipal.add(obj);

obj.toFront();

obj.setLocation(VentanaPrincipal.getWidth() / 2 - obj.getWidth() / 2, VentanaPrincipal.getHeight() / 2 - obj.getHeight() / 2);

try {

obj.setMaximum(false);

} catch (PropertyVetoException ex) {

Logger.getLogger(Interfaz.class.getName()).log(Level.SEVERE, null, ex);

}

obj.setVisible(true);

* 1. **Submenú “Nuevo vendedor”**

CODIGO DEL BOTON “NUEVO VENDEDO”

**Esta parte del código nos permite acceder al submenú “Nuevo vendedor” para poder agregar a algún nuevo vendedor que ingrese a nuestro negocio.**

nuevovendedor objeto = new nuevovendedor();

VentanaPrincipal.add(objeto);

objeto.toFront();

objeto.setLocation(VentanaPrincipal.getWidth() / 2 - objeto.getWidth() / 2, VentanaPrincipal.getHeight() / 2 - objeto.getHeight() / 2);

try {

objeto.setMaximum(false);

} catch (PropertyVetoException ex) {

Logger.getLogger(Stock.Interfaz.class.getName()).log(Level.SEVERE, null, ex);

}

objeto.setVisible(true);

* 1. **Submenú “Generar reporte de venta”**

CODIGO DEL BOTON “GENERAR REPORTE DE VENTA”

**Esta parte del código nos permite acceder al submenú generar reporte de ventas en el cual podemos crear un reporte de las ventas realizadas durante cierto periodo de tiempo.**

reporteventas obj = new reporteventas();

VentanaPrincipal.add(obj);

obj.toFront();

obj.setLocation(VentanaPrincipal.getWidth() / 2 - obj.getWidth() / 2, VentanaPrincipal.getHeight() / 2 - obj.getHeight() / 2);

try {

obj.setMaximum(false);

} catch (PropertyVetoException ex) {

Logger.getLogger(Interfaz.class.getName()).log(Level.SEVERE, null, ex);

}

obj.setVisible(true);

* 1. **Submenú “Editar Stock”**

CODIGO DEL BOTON “EDITAR STOCK”

**Esta parte del código nos permite acceder a la ventana del submenú editar stock para poder así editar el stock que nosotros deseemos.**

verstock objeto = new verstock();

VentanaPrincipal.add(objeto);

objeto.toFront();

objeto.setLocation(VentanaPrincipal.getWidth() / 2 - objeto.getWidth() / 2, VentanaPrincipal.getHeight() / 2 - objeto.getHeight() / 2);

try {

objeto.setMaximum(false);

} catch (PropertyVetoException ex) {

Logger.getLogger(Stock.Interfaz.class.getName()).log(Level.SEVERE, null, ex);

}

objeto.setVisible(true);

* 1. **Submenú “Agregar a Stock”**

CODIGO DEL BOTON “AGREGAR A SROCK”

**Parte del código que nos permite visualizar el submenú agregar a stock para poder guardar un nuevo producto en la base de datos.**

Stock objeto = new Stock();

VentanaPrincipal.add(objeto);

objeto.toFront();

objeto.setLocation(VentanaPrincipal.getWidth() / 2 - objeto.getWidth() / 2, VentanaPrincipal.getHeight() / 2 - objeto.getHeight() / 2);

try {

objeto.setMaximum(false);

} catch (PropertyVetoException ex) {

Logger.getLogger(Stock.Interfaz.class.getName()).log(Level.SEVERE, null, ex);

}

objeto.setVisible(true);

* 1. **Clientes**

CODIGO DEL BOTON “GUARDAR”

**Esta parte del código nos permute guardar un nuevo cliente en la base de datos.**

String vnom, vidcliente, vdire, vtel, campos, valores;

vnom=this.cnom.getText();

vidcliente=this.cid.getText();

vdire=this.cdire.getText();

vtel=this.ctel.getText();

campos="id\_cliente, nom\_cliente, dire\_cliente, tel\_cliente";

valores="'"+vidcliente+"','"+vnom+"','"+vdire+"','"+vtel+"'";

negocliente nc=new negocliente();

nc.guardar("clientes", campos, valores);

limpiar();

guardar.setEnabled(true);

* 1. **Generar venta**

CODIGO DEL BOTON “CALCULAR PRECIO”

**Esta parte del código nos permite hacer la multiplicación de la cantidad de productos comprados por el precio por unidad y calcula el precio total.**

double cantidad;

double precio;

double totalapagar;

cantidad=Double.parseDouble(ccan.getText());

precio=Double.parseDouble(cprecio1.getText());

totalapagar=precio\*cantidad;

totalpagar.setText(" "+totalapagar);

* 1. **Generar factura**

CODIGO DEL BOTON “IMPRIMIR”

**Esta parte del código nos permite imprimir la factura**

try{

PrinterJob job = PrinterJob.getPrinterJob();

job.setPrintable(this);

boolean top = job.printDialog();

if(top){

job.print();

}

} catch (PrinterException pex){

JOptionPane.showMessageDialog(null, "IMPRESION CANCELADA", "Error\n", JOptionPane.INFORMATION\_MESSAGE);

}

CODIGO PARA BUSCAR EL MES

**Esta parte del código nos permite ingresar un mes en la caja de texto para poder buscar el mes que deseemos para asi poder generar nuestro reporte de facturas.**

String[] titulos = {"Cliente", "Cireccion", "Dia", "Mes", "Año", "No. Fac", "Total Fac"};

String[] factura = new String[100];

String sql = "SELECT \* FROM factura WHERE mes LIKE '%" + buscar\_todo.getText() + "%'";

m = new DefaultTableModel(null, titulos);

Conexion conect1 = new Conexion();

con2 = conect1.getConnection();

try

{

st = (Statement) con2.createStatement();

rs = st.executeQuery(sql);

while (rs.next())

{

factura[0] = rs.getString("cliente");

factura[1] = rs.getString("direccion");

factura[2] = rs.getString("dia");

factura[3] = rs.getString("mes");

factura[4] = rs.getString("año");

factura[5] = rs.getString("nf");

factura[6] = rs.getString("totalf");

m.addRow(factura);

}

tabla.setModel(m);

} catch (SQLException ex)

{

System.out.println("ERROR AL BUSCAR LOS DATOS : " + ex.getMessage());

}

* 1. **Nuevo vendedor**

CODIGO DEL BOTON “GUARDAR”

**Parte del código que nos permite guardar un nuevo vendedor en la base de datos, guarda el nuevo usuario y la respectiva contraseña.**

String cnuevouser, ccontra, campos, valores;

cnuevouser=this.cnuevouser.getText();

ccontra=this.ccontra.getText();

campos="Usuario, contraseña";

valores="'"+cnuevouser+"','"+ccontra+"'";

nuevouser n=new nuevouser();

n.guardar("login", campos, valores);

limpiar();

;

guardar.setEnabled(true);

* 1. **Reporte ventas**

CODIGO DEL BOTON “IMPRIMIR”

**Parte del código que nos permite imprimir el reporte de las ventas.**

try

{

PrinterJob job = PrinterJob.getPrinterJob();

job.setPrintable(this);

boolean top = job.printDialog();

if(top)

{

job.print();

}

} catch (PrinterException pex)

{

JOptionPane.showMessageDialog(null, "IMPRESION CANCELADA", "Error\n", JOptionPane.INFORMATION\_MESSAGE);

}

* 1. **Stock**

CODIGO DEL BOTON “NUEVO PRODUCTO”

**Esta parte del código nos habilita los campos de texto para poder escribir en ellos y poder agregar un nuevo producto.**

limpiar();

habilitar();

nuevo.setEnabled(false);

guardar.setEnabled(true);

CODIGO DEL BOTON “GUARDAR”

**Esta parte del código nos permite guardar el nuevo procuro en la base de datos, para ello debemos llenar todos los campos.**

String vcodprod, vnomprod, vex, vpreciop, campos, valores, vcod;

vnomprod=this.nomprod.getText();

vex=this.ex.getText();

vpreciop=this.cpp.getText();

vcod=this.codprod.getText();

campos="Codigo\_producto, Nombre\_producto, Existencia, precio\_publico";

valores="'"+vcod+"','"+vnomprod+"','"+vex+"','"+vpreciop+"'";

negostock ns=new negostock();

ns.guardar("Stock", campos, valores);

limpiar();

desabilitar();

habilitar();

* 1. **Editar Stock**

CODIGO DEL BOTON “EDITAR”

**Esta parte del código nos permite editar el stock que tenemos guardados en la base de datos, para esto debimos guardar algún producto previamente.**

try

{

Connection con = null;

Conexion conect = new Conexion();

con = conect.getConnection();

Statement st = con.createStatement();

String sql = "update Stock set Nombre\_producto = ?, Existencia = ?, Precio\_publico = ? where Codigo\_producto = ?";

PreparedStatement pst = con.prepareStatement(sql);

pst.setString(1, nomprod.getText());

pst.setString(2, ex.getText());

pst.setString(3, cpp.getText());

pst.setInt(4, Integer.parseInt(codprod.getText()));

int n = pst.executeUpdate();

if (n > 0)

{

JOptionPane.showMessageDialog(this, "DATOS ACTUALIZADOS CORRECTAMENTE");

limpiar();

vaciarTabla();

verDatos();

guardar.setEnabled(false);

editar.setEnabled(false);

desabilitar();

}

} catch (SQLException | HeadlessException e)

{

JOptionPane.showMessageDialog(this, "DEBE SELECCIONAR UNA FILA PARA EDITARLA", "Error", JOptionPane.ERROR\_MESSAGE);

}

CODIGO DEL BOTON “GUARDAR”

**Esta parte del código nos permite guardar los cambios generados en el stock.**

if (codprod.getText().isEmpty())

{

JOptionPane.showMessageDialog(this, "¡SOLO SE PUEDE EDITAR!", "Advertencia", JOptionPane.WARNING\_MESSAGE);

codprod.requestFocus();

}

else if (nomprod.getText().isEmpty())

{

JOptionPane.showMessageDialog(this, " ", "Advertencia", JOptionPane.WARNING\_MESSAGE);

nomprod.requestFocus();

}

else if (ex.getText().isEmpty())

{

JOptionPane.showMessageDialog(this, " ", "Advertencia", JOptionPane.WARNING\_MESSAGE);

ex.requestFocus();

}

else if (cpp.getText().isEmpty())

{

JOptionPane.showMessageDialog(this, " ", "Advertencia", JOptionPane.WARNING\_MESSAGE);

cpp.requestFocus();

}

else

{

try

{

Connection con = null;

Conexion conect = new Conexion();

con = conect.getConnection();

Statement st = con.createStatement();

String sql = "insert into Stock (Codigo\_producto,Nombre\_producto,Existencia,precio\_publico) values (?,?,?,?)";

PreparedStatement pst = con.prepareStatement(sql);

pst.setString(1, codprod.getText());

pst.setString(2, nomprod.getText());

pst.setString(3, ex.getText());

pst.setString(4, cpp.getText());

int n = pst.executeUpdate();

if (n > 0)

{

JOptionPane.showMessageDialog(this, "DATOS GUARDADOS CORRECTAMENTE");

vaciarTabla();

guardar.setEnabled(true);

editar.setEnabled(false);

editar.setEnabled(false);

}

} catch (SQLException | HeadlessException e)

{

JOptionPane.showMessageDialog(this, "LOS DATOS NO HAN SIDO GUARDADOS CORRECTAMENTE", "Error", JOptionPane.ERROR\_MESSAGE);

}

}