

Universidad Mariano Gálvez de Guatemala
Facultad de Ingeniería en Sistemas de la Información y Ciencias de la
Comunicación
Programación II
Ing. Jhonny Morales



Tema:
Proyecto Final

Integrantes

Alumno

Carlos Ricardo Hernández Hernández
Luis Fernando José Miguel

Carnet:

0904-21-14883
0904-21-3890

Dany Alexander Ramírez Rivas

0904-21-19064

Hesler Osbaldo Domingo Zacarías

0904-21-19320

Filiberto Castro Bernal

0904-21-20392

Link del Repositorio: [Carlos-Hern/SistemaVentas: Proyecto Final Java Sistema de Ventas \(github.com\)](https://github.com/Carlos-Hern/SistemaVentas)

Índice

Índice.....	2
INTRODUCCION	3
Proceso de Elaboración del Software	4
Descripción de tecnologías	19
Java Swing:.....	19
Conexión a base de datos MYSQL:.....	21
MySQL:	21
Arquitectura Cliente y Servidor:	22
Github:	23
Diagrama de Clases.....	24
.....	24
.....	24
.....	25
Diagrama de Base de Datos	28
Manual de Usuario	29
Conclusiones.....	34

INTRODUCCION

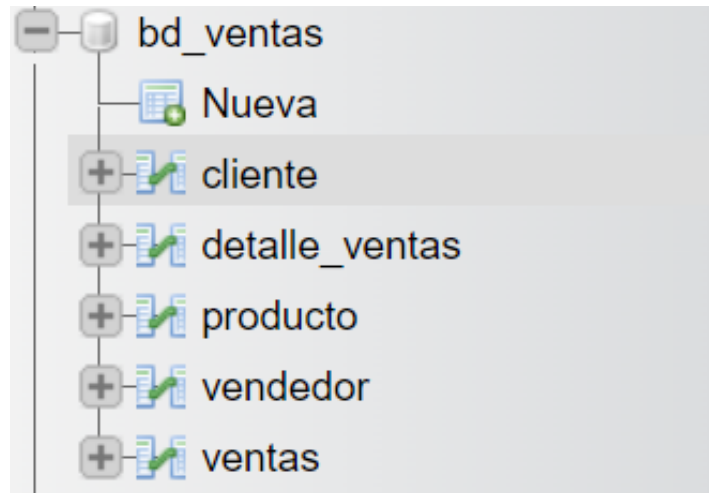
La mayoría de los proyectos actuales necesitan guardar o consultar datos de una base de datos relacional, de esta manera se puede persistir la información de sus usuarios y del negocio que implementan pudiendo acceder a ella en cualquier momento.

Para conectarse a las BD las aplicaciones utilizan por lo general drivers, que no son más que componente que permiten realizar una conexión amigable con la base de datos de un tipo específico. Algunos de estos tipos pueden ser MSSQL, MySQL, Oracle, PostgreSQL, etc.

Otro elemento importante a tener en cuenta es que para realizar operaciones sobre los datos se necesita de un lenguaje de consultas que permita a los desarrolladores trabajar con las bases de datos, y en este punto contamos con SQL.

Proceso de Elaboración del Software

Lo primero que se hizo antes de la elaboración del código, fue hacer la base de datos en MySql, el cual lleva el nombre de bd_ventas, las cuales llevan un total de 5 tablas con sus respectivas variables.



Servidor: 127.0.0.1 » Base de datos: bd_ventas » Tabla: cliente

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Seguimiento

Estructura de tabla Vista de relaciones

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	IdCliente 🔑	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Más
<input type="checkbox"/> 2	Dpi	varchar(13)	utf8mb4_general_ci		Sí	NULL			Cambiar Eliminar Más
<input type="checkbox"/> 3	Nombre	varchar(225)	utf8mb4_general_ci		Sí	NULL			Cambiar Eliminar Más
<input type="checkbox"/> 4	Direccion	varchar(225)	utf8mb4_general_ci		Sí	NULL			Cambiar Eliminar Más
<input type="checkbox"/> 5	Estado	varchar(1)	utf8mb4_general_ci		Sí	NULL			Cambiar Eliminar Más

Servidor: 127.0.0.1 » Base de datos: bd_ventas » Tabla: detalle_ventas

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Seguimiento

Estructura de tabla Vista de relaciones

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	IdDetalleVentas 🔑	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Más
<input type="checkbox"/> 2	IdVentas	int(11)			Sí	NULL			Cambiar Eliminar Más
<input type="checkbox"/> 3	IdProducto	int(11)			Sí	NULL			Cambiar Eliminar Más
<input type="checkbox"/> 4	Cantidad	int(11)			Sí	NULL			Cambiar Eliminar Más
<input type="checkbox"/> 5	PrecioVenta	double			Sí	NULL			Cambiar Eliminar Más

Servidor: 127.0.0.1 » Base de datos: bd_ventas » Tabla: producto

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Seguimiento

Estructura de tabla Vista de relaciones

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	IdProducto	int(11)			No	Ninguna		AUTO_INCREMENT	Má
<input type="checkbox"/> 2	Nombre	varchar(225)	utf8mb4_general_ci		Si	NULL			Má
<input type="checkbox"/> 3	Precio	double			Si	NULL			Má
<input type="checkbox"/> 4	Stock	int(11)			Si	NULL			Má
<input type="checkbox"/> 5	Estado	varchar(1)	utf8mb4_general_ci		Si	NULL			Má

Servidor: 127.0.0.1 » Base de datos: bd_ventas » Tabla: vendedor

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Seguimiento

Estructura de tabla Vista de relaciones

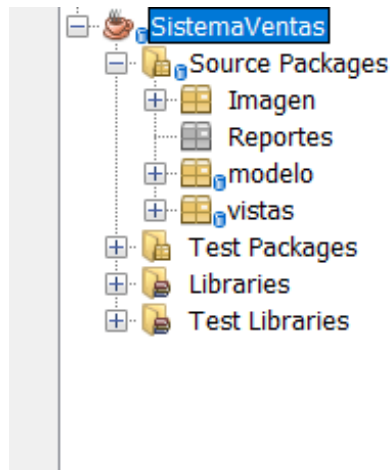
#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	IdVendedor	int(11)			No	Ninguna		AUTO_INCREMENT	Má
<input type="checkbox"/> 2	Dpi	varchar(13)	utf8mb4_general_ci		Si	NULL			Má
<input type="checkbox"/> 3	Nombre	varchar(225)	utf8mb4_general_ci		Si	NULL			Má
<input type="checkbox"/> 4	Telefono	varchar(8)	utf8mb4_general_ci		Si	NULL			Má
<input type="checkbox"/> 5	Estado	varchar(1)	utf8mb4_general_ci		Si	NULL			Má
<input type="checkbox"/> 6	User	varchar(8)	utf8mb4_general_ci		Si	NULL			Má

Servidor: 127.0.0.1 » Base de datos: bd_ventas » Tabla: ventas

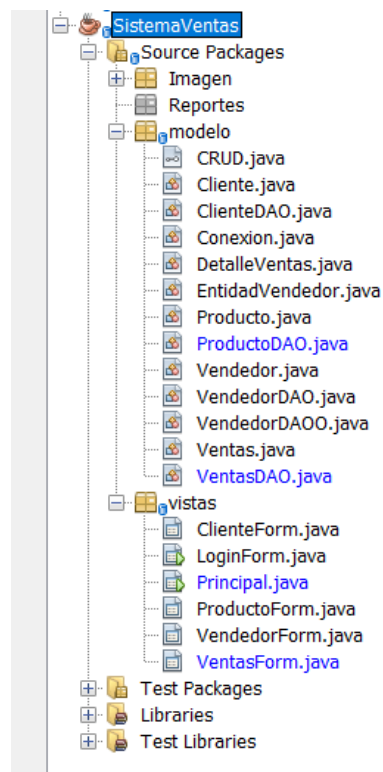
Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Seguimiento

Estructura de tabla Vista de relaciones

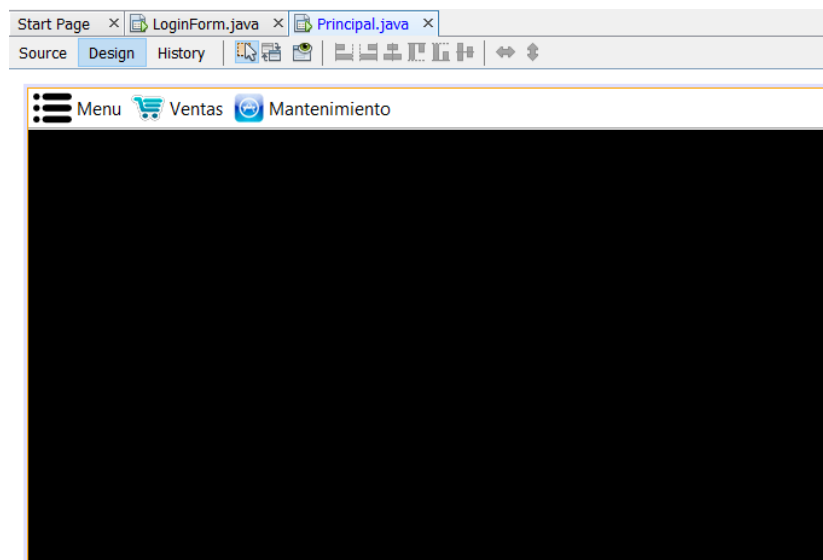
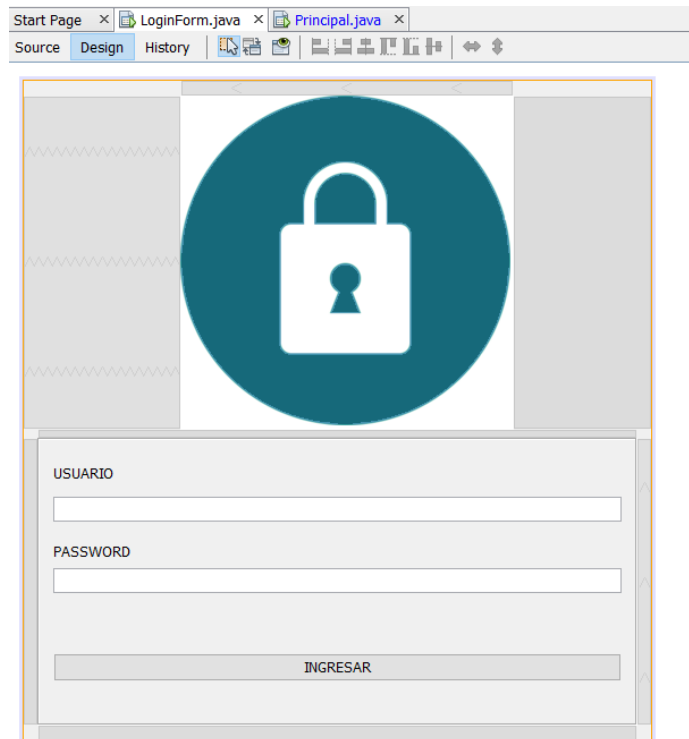
#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	IdVentas	int(11)			No	Ninguna		AUTO_INCREMENT	Má
<input type="checkbox"/> 2	IdCliente	int(11)			Si	NULL			Má
<input type="checkbox"/> 3	IdVendedor	int(11)			Si	NULL			Má
<input type="checkbox"/> 4	NumeroSerie	varchar(225)	utf8mb4_general_ci		Si	NULL			Má
<input type="checkbox"/> 5	FechaVenta	date			Si	current_timestamp()			Má
<input type="checkbox"/> 6	Monto	double			Si	NULL			Má
<input type="checkbox"/> 7	Estado	varchar(1)	utf8mb4_general_ci		Si	NULL			Má



Se creo un nuevo proyecto en el IDE de Netbeans, el cual tiene el nombre de “SistemaVentas”, se le añadió 4 paquetes de Java, de los cuales imagen lleva todo lo que son las imágenes que se utilizaron en el diseño del sistema del proyecto. Modelo y Vistas son los paquetes que más importancia tienen en el proyecto, mientras reportes quedo vacío.



En el paquete de modelo van todas nuestras clases en las cuales van los códigos de cada funcionalidad y en las vistas están los diseños y los códigos en las cuales se conectan las funciones de los modelos y la base de datos de MySQL.



Una vez creadas las vistas y el modelo, se empezó a crear el diseño de principal y login, login sirve para que un empleado pueda entrar al programa, o en pocas palabras, que pueda acceder al diseño principal el cual es donde se generan las ventas, esto conectado en la base de datos de MySQL.

```

Start Page x LoginForm.java x Principal.java x
Source Design History
1 package vistas;
2
3
4
5 import javax.swing.JOptionPane;
6 import modelo.EntidadVendedor;
7 import modelo.VendedorDAO;
8
9
10 public class LoginForm extends javax.swing.JFrame {
11
12     VendedorDAO vdao = new VendedorDAO();
13     EntidadVendedor ev = new EntidadVendedor();
14
15     public LoginForm() {
16         initComponents();
17         this.setLocationRelativeTo(null);
18     }
19
20     /**
21      * This method is called from within the constructor to initialize the form.
22      * WARNING: Do NOT modify this code. The content of this method is always
23      * regenerated by the Form Editor.
24      */
25     @SuppressWarnings("unchecked")
26     Generated Code
112
113     private void btnIngresarActionPerformed(java.awt.event.ActionEvent evt) {
114         validar();
115     }
116
117     public void validar() {
118         String dpi = txtPass.getText();
119         String user = txtUser.getText();
120
121         if (txtUser.getText().equals("") || txtPass.getText().equals("")) {
122             JOptionPane.showMessageDialog(this, "Las cajas estan vacias, debe ingresar datos en las cajas de texto");
123             txtUser.requestFocus();
124         } else {
125             ev = vdao.ValidarVendedor(dpi, user);
126             if (ev.getUser() != null && ev.getDpi() != null) {
127                 Principal p = new Principal();
128                 p.setVisible(true);
129                 dispose();
130             } else {
131                 JOptionPane.showMessageDialog(this, "Debe usar usuarios validos");
132                 txtUser.requestFocus();
133             }
134         }
135     }
136
137     /**

```

```

Start Page x LoginForm.java x Principal.java x
Source Design History
1 package vistas;
2
3
4 import java.awt.Dimension;
5 import javax.swing.JInternalFrame;
6
7
8 public class Principal extends javax.swing.JFrame {
9
10
11     public Principal() {
12         initComponents();
13         this.setExtendedState(MAXIMIZED_BOTH);
14     }
15
16     /**
17      * This method is called from within the constructor to initialize the form.
18      * WARNING: Do NOT modify this code. The content of this method is always
19      * regenerated by the Form Editor.
20      */
21     @SuppressWarnings("unchecked")
22     Generated Code
123
124     private void menuGenerarVentaActionPerformed(java.awt.event.ActionEvent evt) {
125         VentasForm vf = new VentasForm();
126         CentrarVentana(vf);
127     }
128
129     void CentrarVentana(JInternalFrame frame) {
130         VentanaPrincipal.add(frame);
131         Dimension dimension = VentanaPrincipal.getSize();
132         Dimension Dframe = frame.getSize();
133         frame.setLocation((dimension.width - Dframe.height) / 2, (dimension.height - Dframe.width) / 2);
134         frame.show();
135     }
136
137     private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
138         System.exit(0);
139     }
140
141     private void jMenuItem4ActionPerformed(java.awt.event.ActionEvent evt) {
142         ClienteForm cf = new ClienteForm();
143         CentrarVentana(cf);
144     }
145
146     private void jMenuItem5ActionPerformed(java.awt.event.ActionEvent evt) {
147         ProductoForm pf = new ProductoForm();
148         CentrarVentana(pf);
149     }
150
151     private void jMenuItem6ActionPerformed(java.awt.event.ActionEvent evt) {

```

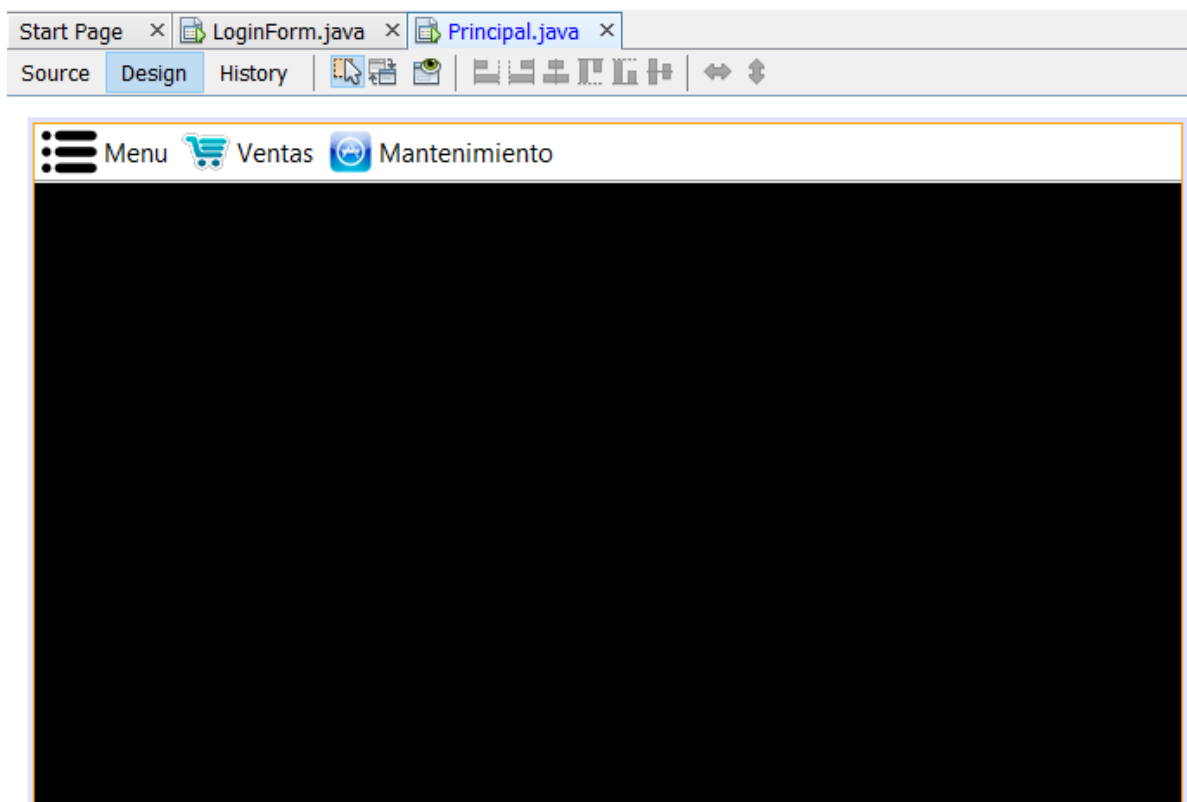


```

148 }
149
150 private void jMenuItem6ActionPerformed(java.awt.event.ActionEvent evt) {
151     VendedorForm vf = new VendedorForm();
152     CentrarVentana(vf);
153 }
154
155 /**
156  * @param args the command line arguments
157  */
158 public static void main(String args[]) {
159     /* Set the Nimbus look and feel */
160     /* Look and feel setting code (optional)
161
162     /* Create and display the form */
163     java.awt.EventQueue.invokeLater(new Runnable() {
164         public void run() {
165             new Principal().setVisible(true);
166         }
167     });
168 }
169
170 // Variables declaration - do not modify
171 public static javax.swing.JDesktopPane VentanaPrincipal;
172 private javax.swing.JMenu jMenuItem1;
173 private javax.swing.JMenu jMenuItem2;
174 private javax.swing.JMenu jMenuItem3;
175 private javax.swing.JMenuBar jMenuItemBar1;
176 private javax.swing.JMenuItem jMenuItem1;
177 private javax.swing.JMenuItem jMenuItem2;
178 private javax.swing.JMenuItem jMenuItem4;
179 private javax.swing.JMenuItem jMenuItem5;
180 private javax.swing.JMenuItem jMenuItem6;
181 private javax.swing.JMenuItem jMenuItemGenerarVenta;
182 // End of variables declaration
183

```

Se genera el código de Login y Principal



En el diseño de principal hay tres opciones, una es menú que contiene la opción de ayuda/salir y las otras dos son ventas y mantenimiento, cada una de ellas a excepción de menú tienen JInternalFrame Form, las cuales son las siguientes:

The screenshot shows a Java Swing application window titled "Sistema de Ventas". The window contains a header with a shopping basket icon and the text "PUNTO DE VENTA 'El Abarrotero'", "Venta de Abarros", and "Tel: 24280000". Below the header is a form with several input fields and buttons. The form is organized into sections for client and product information, price, and quantity. At the bottom of the form is a table with columns for item number, code, product name, unit price, quantity, and total. Below the table is a section for generating a sale and displaying the total amount to be paid.

Para ventas está generar venta, la cual lleva a esta ventana llamada VentasForm, la cual almacenará los datos de producto, y clientes, cada venta que se realice se almacenará en la base de datos en el apartado de ventas y detalle_ventas.

```

Start Page x Principal.java x ClienteForm.java x ProductoForm.java x VendedorForm.java x VentasForm.java x
Source Design History
1 package vistas;
2
3
4 import java.util.ArrayList;
5 import java.util.Calendar;
6 import java.util.GregorianCalendar;
7 import javax.swing.JOptionPane;
8 import javax.swing.table.DefaultTableModel;
9 import modelo.Cliente;
10 import modelo.ClienteDAO;
11 import modelo.DetalleVentas;
12 import modelo.Producto;
13 import modelo.ProductoDAO;
14 import modelo.Ventas;
15 import modelo.VentasDAO;
16
17
18 public class VentasForm extends javax.swing.JInternalFrame {
19
20     VentasDAO vdao = new VentasDAO();
21     ClienteDAO cdao = new ClienteDAO();
22     ProductoDAO pdao = new ProductoDAO();
23     Producto p = new Producto();
24     Ventas v = new Ventas();
25     DetalleVentas dv = new DetalleVentas();
26
27     Cliente cliente = new Cliente();
28
29     DefaultTableModel modelo = new DefaultTableModel();
30     int idp;
31     int cant;
32     double pre;
33     double tpagar;
34
35     public VentasForm() {
36         initComponents();
37         Calendar calendar = new GregorianCalendar();
38         txtFecha.setText(""+calendar.get(Calendar.YEAR)+"-"+calendar.get(Calendar.MONTH)+"-"+calendar.get(Calendar.DAY_OF_MONTH));
39     }
40
41     /**
42      * This method is called from within the constructor to initialize the form.
43      * WARNING: Do NOT modify this code. The content of this method is always
44      * regenerated by the Form Editor.
45      */
46     @SuppressWarnings("unchecked")
47     // Generated Code
48
49
50     private void txtClienteActionPerformed(java.awt.event.ActionEvent evt) {
51         // TODO add your handling code here:
52     }
53 }

```

```

Start Page x Principal.java x ClienteForm.java x ProductoForm.java x VendedorForm.java x VentasForm.java x
Source Design History
372 private void txtClienteActionPerformed(java.awt.event.ActionEvent evt) {
373     // TODO add your handling code here:
374 }
375
376 private void btnBuscarClienteActionPerformed(java.awt.event.ActionEvent evt) {
377     buscarCliente();
378 }
379
380 private void btnBuscarProductoActionPerformed(java.awt.event.ActionEvent evt) {
381     buscarProducto();
382 }
383
384 private void btnAgregarActionPerformed(java.awt.event.ActionEvent evt) {
385     agregarProducto();
386 }
387
388 private void btnGenerarVentaActionPerformed(java.awt.event.ActionEvent evt) {
389     guardarVenta();
390     guardarDetalle();
391 }
392
393 void guardarVenta(){
394     int idv = 1;
395     int idc = cliente.getId();
396     String serie=txtSerie.getText();
397     String fecha=txtFecha.getText();
398     double monto=tpagar;
399     String estado = "1";
400     v.setIdCliente(idc);
401     v.setIdVendedor(idv);
402     v.setSerie(serie);
403     v.setFecha(fecha);
404     v.setMonto(monto);
405     v.setEstado(estado);
406     vdao.GuardarVentas(v);
407 }
408
409 void guardarDetalle(){
410     String idv = vdao.IdVentas();
411     int idve=Integer.parseInt(idv);
412     for(int i = 0; i < TablaDetalle.getRowCount(); i++){
413         int idp = Integer.parseInt(TablaDetalle.getValueAt(i, 1).toString());
414         int cant = Integer.parseInt(TablaDetalle.getValueAt(i, 3).toString());
415         double pre = Double.parseDouble(TablaDetalle.getValueAt(i, 4).toString());
416         dv.setIdVentas(idve);
417         dv.setIdProducto(idp);
418         dv.setCantidad(cant);
419         dv.setPreVenta(pre);
420         vdao.GuardarDetalleVentas(dv);
421     }
422 }

```

```

Start Page x Principal.java x ClienteForm.java x ProductoForm.java x VendedorForm.java x VentasForm.java x
Source Design History
421 }
422 }
423 void agregarProducto() {
424     double total;
425     int item = 0;
426     modelo = (DefaultTableModel)TablaDetalle.getModel();
427     item = item+1;
428     idp=p.getId();
429     String nomp = txtProducto.getText();
430     pre =Double.parseDouble(txtPrecio.getText());
431     cant=Integer.parseInt(txtCantidad.getValue().toString());
432     int stock=Integer.parseInt(txtStock.getText());
433     total = cant*pre;
434     ArrayList lista = new ArrayList();
435     if (stock > 0){
436         lista.add(item);
437         lista.add(idp);
438         lista.add(nomp);
439         lista.add(pre);
440         lista.add(cant);
441         lista.add(total);
442         Object[] ob = new Object[6];
443
444         ob[0] = lista.get(0);
445         ob[1] = lista.get(1);
446         ob[2] = lista.get(2);
447         ob[3] = lista.get(3);
448         ob[4] = lista.get(4);
449         ob[5] = lista.get(5);
450         modelo.addRow(ob);
451         TablaDetalle.setModel(modelo);
452
453         calculaTotal();
454     }else{
455         JOptionPane.showMessageDialog(this, "Stock Producto no Disponible");
456     }
457 }
458 void calculaTotal() {
459     tpagar=0;
460     for (int i = 0; i < TablaDetalle.getRowCount(); i++){
461         cant = Integer.parseInt(TablaDetalle.getValueAt(i, 4).toString());
462         pre = Double.parseDouble(TablaDetalle.getValueAt(i, 5).toString());
463         tpagar=tpagar+(cant*pre);
464     }
465     txtTotalPagar.setText(""+tpagar+"0");
466 }
467 void buscarProducto() {
468     int id=Integer.parseInt(txtCodProducto.getText());
469     if(txtCodProducto.getText().equals("")){
470         JOptionPane.showMessageDialog(this, "Debe ingresar el codigo del Producto");
471     }
472 }
473 void buscarCliente() {
474     int r = 0;
475     String cod=txtCodCliente.getText();
476     if(txtCodCliente.getText().equals("")){
477         JOptionPane.showMessageDialog(this, "Debe ingresar codigo Cliente");
478     }else{
479         cliente = cdao.listarID(cod);
480         if(cliente.getDpi() !=null){
481             txtCliente.setText(cliente.getNom());
482             txtCodProducto.requestFocus();
483         }else{
484             JOptionPane.showMessageDialog(this, "Cliente no registrado, Desea Registrar?");
485             if(r==0){
486                 ClienteForm cf=new ClienteForm();
487                 Principal.VentanaPrincipal.add(cf);
488                 cf.setVisible(true);
489             }
490         }
491     }
492 }
493
494 // Variables declaration - do not modify
495 private javax.swing.JTable TablaDetalle;
496 private javax.swing.JButton btnAgregar;
497 private javax.swing.JButton btnBuscarCliente;
498 private javax.swing.JButton btnBuscarProducto;
499 private javax.swing.JButton btnGenerarVenta;
500 private javax.swing.JLabel jLabel1;
501 private javax.swing.JLabel jLabel10;
502 private javax.swing.JLabel jLabel11;
503 private javax.swing.JLabel jLabel12;

```

Código del form de generar ventas, está instanciado con cada una de las clases del proyecto para su funcionalidad.

Después se generó el Mantenimiento el cual lleva dentro de ella Clientes, Producto y Vendedor, cada uno de ellos se almacena en la base de datos de MySql en las tablas cliente, producto y vendedor.

La primera tabla es de cliente y en el programa es ClienteForm, en este apartado se podrá ingresar, actualizar, eliminar o asignar un nuevo cliente, el cual va entrelazado en la parte generar ventas y también tiene su apartado en la base de datos de MySql.

```

1 package vistas;
2
3
4 import java.util.List;
5 import javax.swing.JOptionPane;
6 import javax.swing.table.DefaultTableModel;
7 import modelo.Cliente;
8 import modelo.ClienteDAO;
9
10
11 public class ClienteForm extends javax.swing.JInternalFrame {
12     ClienteDAO dao = new ClienteDAO();
13     Cliente cl = new Cliente();
14
15     DefaultTableModel modelo = new DefaultTableModel();
16     int id;
17     public ClienteForm() {
18         initComponents();
19         listar();
20     }
21     void listar(){
22         List<Cliente> lista = dao.listar();
23         modelo=(DefaultTableModel)tabla.getModel();
24         Object[] ob= new Object[5];
25
26         for(int i = 0; i < lista.size(); i++){
27             ob[0]=lista.get(i).getId();
28             ob[1]=lista.get(i).getDpi();
29             ob[2]=lista.get(i).getNom();
30             ob[3]=lista.get(i).getDir();
31             ob[4]=lista.get(i).getEstado();
32             modelo.addRow(ob);
33         }
34         tabla.setModel(modelo);
35     }
36
37
38
39 @SuppressWarnings("unchecked")
40 // Generated Code
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
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
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
2
```

```

Start Page x Principal.java x ClienteForm.java x ProductoForm.java x VendedorForm.java x
Source Design History | [Icons]
223 nuevo();
224 }
225
226 private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt) {
227     actualizar();
228     limpiarTabla();
229     listar();
230     nuevo();
231 }
232
233 private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
234     eliminar();
235     limpiarTabla();
236     listar();
237     nuevo();
238 }
239
240 private void btnNuevoActionPerformed(java.awt.event.ActionEvent evt) {
241     nuevo();
242 }
243
244 private void tablaMouseClicked(java.awt.event.MouseEvent evt) {
245     int fila = tabla.getSelectedRow();
246     if (fila==1) {
247         JOptionPane.showMessageDialog(this, "Debe Seleccionar una Fila");
248     }else{
249         id = Integer.parseInt(tabla.getValueAt(fila, 0).toString());
250         String dpi = tabla.getValueAt(fila,1).toString();
251         String nom = tabla.getValueAt(fila,2).toString();
252         String dir = tabla.getValueAt(fila,3).toString();
253         String est = tabla.getValueAt(fila,4).toString();
254
255         txtDpi.setText(dpi);
256         txtNombres.setText(nom);
257         txtDireccion.setText(dir);
258         cboEstado.setSelectedItem(est);
259     }
260 }
261
262 void agregar() {
263     String dpi = txtDpi.getText();
264     String nom = txtNombres.getText();
265     String dir = txtDireccion.getText();
266     String est = cboEstado.getSelectedItem().toString();
267
268     Object[] ob = new Object[4];
269     ob[0] = dpi;
270     ob[1] = nom;
271     ob[2] = dir;
272     ob[3] = est;
273     dao.add(ob);
274 }
275
276 void actualizar() {
277     int fila = tabla.getSelectedRow();
278     if (fila==1) {
279         JOptionPane.showMessageDialog(this, "Debe seleccionar una fila");
280     }else{
281         String dpi = txtDpi.getText();
282         String nom = txtNombres.getText();
283         String dir = txtDireccion.getText();
284         String est = cboEstado.getSelectedItem().toString();
285
286         Object[] obj = new Object[5];
287         obj[0] = dpi;
288         obj[1] = nom;
289         obj[2] = dir;
290         obj[3] = est;
291         obj[4] = id;
292         dao.actualizar(obj);
293     }
294 }
295
296 void eliminar() {
297     int fila=tabla.getSelectedRow();
298     int ids = Integer.parseInt(tabla.getValueAt(fila, 0).toString());
299     if (fila==1) {
300         JOptionPane.showMessageDialog(this, "Debe seleccionar una fila");
301     }else{
302         dao.eliminar(ids);
303     }
304 }
305
306 void nuevo() {
307     txtNombres.setText("");
308     txtDireccion.setText("");
309     txtDpi.setText("");
310     cboEstado.setSelectedItem(0);
311     txtDpi.requestFocus();
312 }
313
314 void limpiarTabla() {
315     for(int i = 0; i < modelo.getRowCount(); i++){
316         modelo.removeRow(i);
317         i = i-1;
318     }
319 }
320
321 // Variables declaration - do not modify
322 private javax.swing.JButton btnActualizar;
323 private javax.swing.JButton btnAgregar;
324 private javax.swing.JButton btnEliminar;
325 private javax.swing.JButton btnNuevo;
326 private javax.swing.JComboBox<String> cboEstado;

```

Este es el código de ClienteForm, va instanciado con las clases Cliente y ClienteDAO, el cual es donde se conecta y almacena directamente de la tabla clientes en MySQL.

El segundo Form es de productos tanto en MySQL como en el programa, en este apartado se puede agregar, actualizar, eliminar y nuevo producto, el cual se irá asociando a la base de datos producto.

```

1 package vistas;
2
3
4 import java.util.List;
5 import javax.swing.JOptionPane;
6 import javax.swing.table.DefaultTableModel;
7 import modelo.Producto;
8 import modelo.ProductoDAO;
9
10
11
12 public class ProductoForm extends javax.swing.JInternalFrame {
13     ProductoDAO dao= new ProductoDAO();
14     Producto cl = new Producto();
15
16     DefaultTableModel modelo =new DefaultTableModel();
17     int id;
18     public ProductoForm() {
19         initComponents();
20         listar();
21     }
22     void listar(){
23         List<Producto> lista=dao.listar();
24         modelo=(DefaultTableModel)table.getModel();
25         Object[] ob= new Object[5];
26
27         for(int i = 0; i < lista.size(); i++){
28             ob[0]=lista.get(i).getId();
29             ob[1]=lista.get(i).getNom();
30             ob[2]=lista.get(i).getPrecio();
31             ob[3]=lista.get(i).getCant();
32             ob[4]=lista.get(i).getEst();
33             modelo.addRow(ob);
34         }
35         table.setModel(modelo);
36     }
37     /**
38      * This method is called from within the constructor to initialize the form.
39      * WARNING: Do NOT modify this code. The content of this method is always
40      * regenerated by the Form Editor.
41      */
42     @SuppressWarnings("unchecked")
43     Generated Code
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
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
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

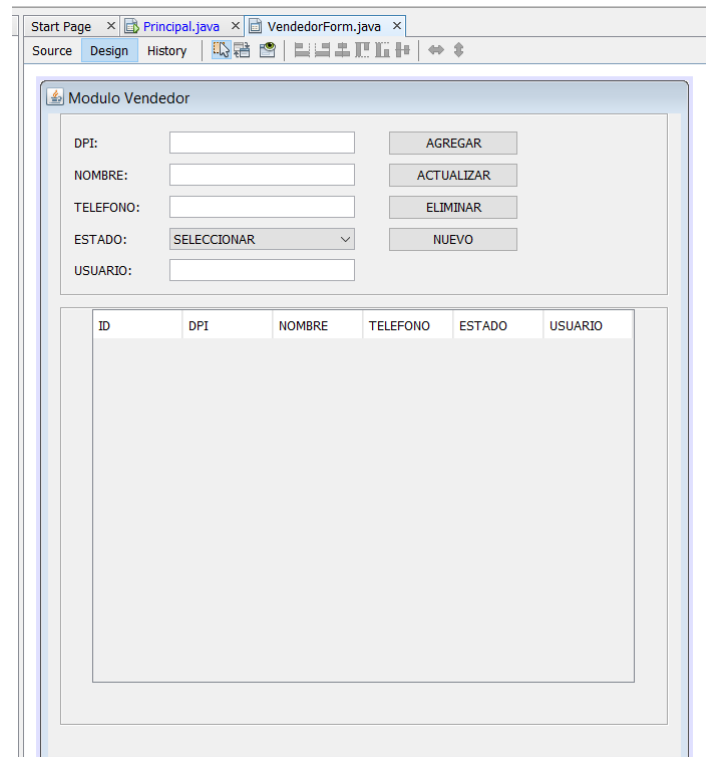
```

```

Start Page x Principal.java x ProductoForm.java x VendedorForm.java x
Source Design History
225 limpiarTabla();
226 listar();
227
228 }
229
230 private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt) {
231     actualizar();
232     limpiarTabla();
233     listar();
234 }
235
236 private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
237     eliminar();
238     limpiarTabla();
239     listar();
240 }
241
242 private void btnNuevoActionPerformed(java.awt.event.ActionEvent evt) {
243     nuevo();
244 }
245
246 private void tablaMouseClicked(java.awt.event.MouseEvent evt) {
247     int fila = tabla.getSelectedRow();
248     if (fila== -1) {
249
250     }else{
251         id= Integer.parseInt(tabla.getValueAt(fila, 0).toString());
252         String nom = tabla.getValueAt(fila, 1).toString();
253         String precio = tabla.getValueAt(fila, 2).toString();
254         String cant = tabla.getValueAt(fila, 3).toString();
255         String est = tabla.getValueAt(fila, 4).toString();
256
257         txtNombre.setText(nom);
258         txtPrecio.setText(precio);
259         txtCantidad.setText(cant);
260         cboEstado.setSelectedItem(est);
261     }
262 }
263
264 void agregar() {
265     String nom = txtNombre.getText();
266     String precio = txtPrecio.getText();
267     String cant = txtCantidad.getText();
268     String est = cboEstado.getSelectedItem().toString();
269     Object [] ob = new Object[4];
270     ob[0]=nom;
271     ob[1]=precio;
272     ob[2]=cant;
273     ob[3]=est;
274     dao.add(ob);
275
276     ob[3]=est;
277     dao.add(ob);
278 }
279 void actualizar() {
280     int fila = tabla.getSelectedRow();
281     if (fila== -1) {
282         JOptionPane.showMessageDialog(this, "Debe seleccionar una fila");
283     }else{
284         String nom = txtNombre.getText();
285         String precio = txtPrecio.getText();
286         String cant = txtCantidad.getText();
287         String est = cboEstado.getSelectedItem().toString();
288
289         Object[] obj = new Object[5];
290         obj[0]=id;
291         obj[1]=nom;
292         obj[2]=precio;
293         obj[3]=cant;
294         obj[4]=est;
295
296         dao.actualizar(obj);
297     }
298 }
299 void eliminar() {
300     int fila=tabla.getSelectedRow();
301     int ids = Integer.parseInt(tabla.getValueAt(fila, 0).toString());
302     if (fila== -1) {
303         JOptionPane.showMessageDialog(this, "Debe seleccionar una fila");
304     }else{
305         dao.eliminar(ids);
306     }
307 }
308 void nuevo() {
309     txtNombre.setText("");
310     txtPrecio.setText("");
311     txtCantidad.setText("");
312     cboEstado.setSelectedItem(0);
313 }
314 void limpiarTabla() {
315     for(int i = 0; i < modelo.getRowCount(); i++){
316         modelo.removeRow(i);
317         i = i-1;
318     }
319 }
320
321 // Variables declaration - do not modify
322 private javax.swing.JButton btnActualizar;
323 private javax.swing.JButton btnAgregar;

```

Este es el código de ProductoForm, va instanciado con las clases Producto y ProductoDAO, el cual es donde se conecta y almacena directamente de la tabla producto en MySQL.



Por último se tiene el Form de vendedor, el cual sirve para almacenar los datos de los vendedores en la Base de datos de MySql en la tabla vendedor, en este se puede agregar, actualizar, eliminar y agregar un nuevo vendedor, recordar que el vendedor también es un usuario para el login.

```

1 package vistas;
2
3
4 import java.util.List;
5 import javax.swing.JOptionPane;
6 import javax.swing.table.DefaultTableModel;
7 import modelo.Vendedor;
8 import modelo.VendedorDAO;
9
10
11 public class VendedorForm extends javax.swing.JInternalFrame {
12     VendedorDAO dao= new VendedorDAO();
13     Vendedor cl = new Vendedor();
14
15     DefaultTableModel modelo = new DefaultTableModel();
16     int id;
17     public VendedorForm() {
18         initComponents();
19         listar();
20     }
21
22     void listar(){
23         List<Vendedor> lista=dao.listar();
24         modelo=(DefaultTableModel)tabla.getModel();
25         Object[] ob= new Object[6];
26         for(int i = 0; i < lista.size(); i++){
27             ob[0]=lista.get(i).getId();
28             ob[1]=lista.get(i).getDpi();
29             ob[2]=lista.get(i).getNom();
30             ob[3]=lista.get(i).getTel();
31             ob[4]=lista.get(i).getEst();
32             ob[5]=lista.get(i).getUser();
33             modelo.addRow(ob);
34         }
35         tabla.setModel(modelo);
36     }
37
38     /**
39      * This method is called from within the constructor to initialize the form.
40      * WARNING: Do NOT modify this code. The content of this method is always
41      * regenerated by the Form Editor.
42      */
43     @SuppressWarnings("unchecked")
44     Generated Code
45
46
47     private void btnAgregarActionPerformed(java.awt.event.ActionEvent evt) {
48         agregar();
49         limpiarTabla();
50         listar();
51     }

```

```

Start Page x Principal.java x VendedorForm.java x
Source Design History
228 private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt) {
229     actualizar();
230     limpiarTabla();
231     listar();
232 }
233
234 private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
235     eliminar();
236     limpiarTabla();
237     listar();
238 }
239
240 private void btnNuevoActionPerformed(java.awt.event.ActionEvent evt) {
241     nuevo();
242 }
243
244 private void tablaMouseClicked(java.awt.event.MouseEvent evt) {
245     int fila = tabla.getSelectedRow();
246     if (fila == -1) {
247         JOptionPane.showMessageDialog(this, "Debe Seleccionar una Fila");
248     }
249     else {
250         id = Integer.parseInt(tabla.getValueAt(fila, 0).toString());
251         String dpi = tabla.getValueAt(fila, 1).toString();
252         String nom = tabla.getValueAt(fila, 2).toString();
253         String tel = tabla.getValueAt(fila, 3).toString();
254         String est = tabla.getValueAt(fila, 4).toString();
255         String user = tabla.getValueAt(fila, 5).toString();
256
257         txtDpi.setText(dpi);
258         txtNombre.setText(nom);
259         txtTelefono.setText(tel);
260         cboEstado.setSelectedItem(est);
261         txtUsuario.setText(user);
262     }
263 }
264
265 void agregar() {
266     String dpi = txtDpi.getText();
267     String nom = txtNombre.getText();
268     String tel = txtTelefono.getText();
269     String est = cboEstado.getSelectedItem().toString();
270     String user = txtUsuario.getText();
271     Object [] ob = new Object[5];
272     ob[0] = dpi;
273     ob[1] = nom;
274     ob[2] = tel;
275     ob[3] = est;
276     ob[4] = user;
277
278     dao.add(ob);
279 }
280 void actualizar() {
281     int fila = tabla.getSelectedRow();
282     if (fila == -1) {
283         JOptionPane.showMessageDialog(this, "Debe seleccionar una fila");
284     }
285     else {
286         String dpi = txtDpi.getText();
287         String nom = txtNombre.getText();
288         String tel = txtTelefono.getText();
289         String est = cboEstado.getSelectedItem().toString();
290         String user = txtUsuario.getText();
291         Object [] obj = new Object[6];
292         obj[0] = id;
293         obj[1] = dpi;
294         obj[2] = nom;
295         obj[3] = tel;
296         obj[4] = est;
297         obj[5] = user;
298         dao.actualizar(obj);
299     }
300 }
301 void eliminar() {
302     int fila = tabla.getSelectedRow();
303     int ids = Integer.parseInt(tabla.getValueAt(fila, 0).toString());
304     if (fila == -1) {
305         JOptionPane.showMessageDialog(this, "Debe seleccionar una fila");
306     }
307     else {
308         dao.eliminar(ids);
309     }
310 }
311 void nuevo() {
312     txtNombre.setText("");
313     txtTelefono.setText("");
314     txtDpi.setText("");
315     cboEstado.setSelectedItem(0);
316     txtUsuario.setText("");
317     txtDpi.requestFocus();
318 }
319 void limpiarTabla() {
320     for(int i = 0; i < modelo.getRowCount(); i++) {
321         modelo.removeRow(i);
322         i = i - 1;
323     }
324 }
325 // Variables declaration - do not modify
326 private javax.swing.JButton btnActualizar;
327 private javax.swing.JButton btnAgregar;

```

Este es el código de VendedorForm, va instanciado con las clases Vendedor, VendedorDAO y VendedorDAO, el cual es donde se conecta y almacena directamente de la tabla vendedor en MySql.

Descripción de tecnologías

Java Swing:

Java Swing es un kit de herramientas de widgets de interfaz gráfica de usuario de Java ligero que incluye un amplio conjunto de widgets. Forma parte de Java Foundation Classes e incluye varios paquetes para desarrollar aplicaciones de escritorio enriquecidas en Java. Swing incluye controles integrados como árboles, botones de imagen, paneles con pestañas, controles deslizantes, barras de herramientas, selectores de color, tablas y áreas de texto para mostrar HTTP o formato de texto enriquecido. Los componentes Swing están escritos completamente en Java y, por lo tanto, son independientes de la plataforma. **Swing** también es un paquete o librería de Java que **sirve para crear aplicaciones gráficas** que permiten la interacción con el usuario. Está disponible desde la versión 1.2 de Java.

Incluye todos los recursos necesarios para construir interfaces gráficas, algunos de estos los ha heredado de su predecesor AWT, pero adiciona cosas nuevas, que corrigen algunas dificultades que se presentaban con anterioridad.

Su uso puede resultar complejo, pero su gran ventaja es que el código desarrollado tiene la posibilidad de **ejecutarse en cualquier sistema operativo**, manteniendo la portabilidad que es una característica que Java continúa ofreciendo a sus usuarios.

Para poder usar Swing es necesario importar el paquete javax.swing, que tiene clases como J

Menu, JRadioButton, JCheckbox, JButton, JTextField, etc. que le otorgan funcionalidad a las aplicaciones que se programen con esta librería.

Las ventanas son el elemento fundamental en el trabajo con Swing y tienen ciertas características como:

- Al generar una ventana o frame será necesario emplear el método setVisible para que sea posible verla.
- El tamaño es estándar y debe ser ajustado mediante el método setSize.
- Es necesario programar el comportamiento que tendrá el programa cuando se cierren las ventanas, puede cerrarse completamente o esconder las ventanas, según la necesidad.

Ahora bien, una característica que hace sumamente atractivo el trabajo con esto, es el hecho de que se puede trabajar desde **una vista de diseño** que facilita la tarea de hacer interfaces gráficas.

Java orientada a objetos: La primera característica, orientado a objetos, se refiere a un método de programación y al diseño del lenguaje. Aunque hay muchas interpretaciones para, una primera idea es diseñar el software de forma que los distintos tipos de datos que usen estén unidos a sus operaciones. Así, los datos y el código funciones o métodos se combinan en entidades llamadas objetos. Un objeto puede verse como un paquete que contiene el comportamiento el código y el estado datos. El principio es separar aquello que cambia de las cosas que permanecen inalterables. Frecuentemente, cambiar una estructura de datos implica un cambio en el código que opera sobre los mismos, o viceversa. Esta separación en objetos coherentes e independientes ofrece una base más estable para el diseño de un sistema software. El objetivo es hacer que grandes proyectos sean fáciles de gestionar y manejar, mejorando como consecuencia su calidad y reduciendo el número de proyectos fallidos.

Otra de las grandes promesas de la programación orientada a objetos es la creación de entidades más genéricas objetos que permitan la reutilización del software entre proyectos, una de las premisas fundamentales de la Ingeniería del Software. Un objeto genérico, por ejemplo, debería en teoría tener el mismo conjunto de comportamiento en diferentes proyectos, sobre todo cuando estos coinciden en cierta medida, algo que suele suceder en las grandes organizaciones. En este sentido, los objetos podrían verse como piezas reutilizables que pueden emplearse en múltiples proyectos distintos, posibilitando así a la industria del software construir proyectos de envergadura empleando componentes ya existentes y de comprobada calidad, conduciendo esto finalmente a una reducción drástica del tiempo de desarrollo. Podemos usar como ejemplo de objeto el aluminio. Una vez definidos datos (densidad, maleabilidad, etc.), y su “comportamiento” (soldar dos piezas, etc.), el objeto “aluminio” puede ser reutilizado en el campo de la construcción, del automóvil, de la aviación, etc.

La reutilización del software ha experimentado resultados dispares, encontrando dos dificultades principales: el diseño de objetos realmente genéricos es pobremente comprendido, y falta una metodología para la amplia comunicación de oportunidades de reutilización. Algunas comunidades de “código abierto” (open source) quieren ayudar en este problema dando medios a los desarrolladores para diseminar la información sobre el uso y versatilidad de objetos reutilizables y bibliotecas de objetos.

Conexión a base de datos MYSQL:

En programación es prácticamente inevitable trabajar con algún tipo de sistema de gestión de bases de datos. Cualquier programa que imaginemos tarde o temprano necesitará almacenar datos en algún lugar, como mínimo para poder almacenar la lista de usuarios autorizados, sus permisos y propiedades.

MySQL:

Es un sistema de gestión de bases de datos que cuenta con una doble licencia. Por una parte es de código abierto, pero por otra, cuenta con una versión comercial gestionada por la compañía Oracle. Actualmente, es la base de datos de código abierto más famosa y utilizada en el mundo entero.

Como él, podemos encontrar otras como la propia Oracle o Microsoft SQL Server. Todas tienen la misma finalidad y se utilizan en el mismo entorno, que no es más que el desarrollo web, y son las que más se utilizan actualmente para dar forma y facilitar la comunicación entre webs y servidores.

Una de las principales características de MySQL es que trabaja con bases de datos relacionales, es decir, utiliza tablas múltiples que se interconectan entre sí para almacenar la información y organizarla correctamente. A pesar de su finalidad y del entorno en el que se utiliza, cabe destacar que es un sistema desarrollado originalmente en C y C++, uno de los lenguajes de programación más tradicionales y antiguos que existen.

Con actualizaciones constantes y el gran punto a favor de ser un entorno gratuito y perfectamente modificable, MySQL se ha ganado la posición como uno de los más usados en el sector digital. La inmensa mayoría de programadores que trabajan en desarrollo web han pasado por el uso de esta herramienta por sus capacidades y sus bondades.

.

Las versiones Enterprise, diseñadas para aquellas empresas que quieran incorporarlo en productos privativos, incluyen productos o servicios adicionales tales como herramientas de monitorización y asistencia técnica oficial.

MySQL presenta algunas ventajas que lo hacen muy interesante para los desarrolladores. La más evidente es que trabaja con bases de datos relacionales, es decir, utiliza tablas múltiples que se interconectan entre sí para almacenar la información y organizarla correctamente.

Al ser basada en código abierto es fácilmente accesible y la inmensa mayoría de programadores que trabajan en desarrollo web han pasado usar MySQL en alguno de

sus proyectos porque al estar ampliamente extendido cuenta además con una ingente comunidad que ofrece soporte a otros usuarios. Pero estas no son las únicas características como veremos a continuación:

Arquitectura Cliente y Servidor:

MySQL basa su funcionamiento en un modelo cliente y servidor. Es decir, clientes y servidores se comunican entre sí de manera diferenciada para un mejor rendimiento. Cada cliente puede hacer consultas a través del sistema de registro para obtener datos, modificarlos, guardar estos cambios o establecer nuevas tablas de registros, por ejemplo.

Compatibilidad con SQL: SQL es un lenguaje generalizado dentro de la industria. Al ser un estándar MySQL ofrece plena compatibilidad por lo que si has trabajado en otro motor de bases de datos no tendrás problemas en migrar a MySQL.

Vistas: Desde la versión 5.0 de MySQL se ofrece compatibilidad para poder configurar vistas personalizadas del mismo modo que podemos hacerlo en otras bases de datos SQL. En bases de datos de gran tamaño las vistas se hacen un recurso imprescindible.

Procedimientos almacenados. MySQL posee la característica de no procesar las tablas directamente sino que a través de procedimientos almacenados es posible incrementar la eficacia de nuestra implementación.

Desencadenantes. MySQL permite además poder automatizar ciertas tareas dentro de nuestra base de datos. En el momento que se produce un evento otro es lanzado para actualizar registros u optimizar su funcionalidad.

Transacciones. Una transacción representa la actuación de diversas operaciones en la base de datos como un dispositivo. El sistema de base de registros avala que todos los procedimientos se establezcan correctamente o ninguna de ellas. En caso por ejemplo de una falla de energía, cuando el monitor falla u ocurre algún otro inconveniente, el sistema opta por preservar la integridad de la base de datos resguardando la información.

Github:

GitHub es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control. Se utiliza principalmente para la creación de código fuente de programas de ordenador. GitHub opera bajo el nombre de *GitHub*, como Logical Awesome LLC. El código de los proyectos alojados en GitHub se almacena generalmente de forma pública.

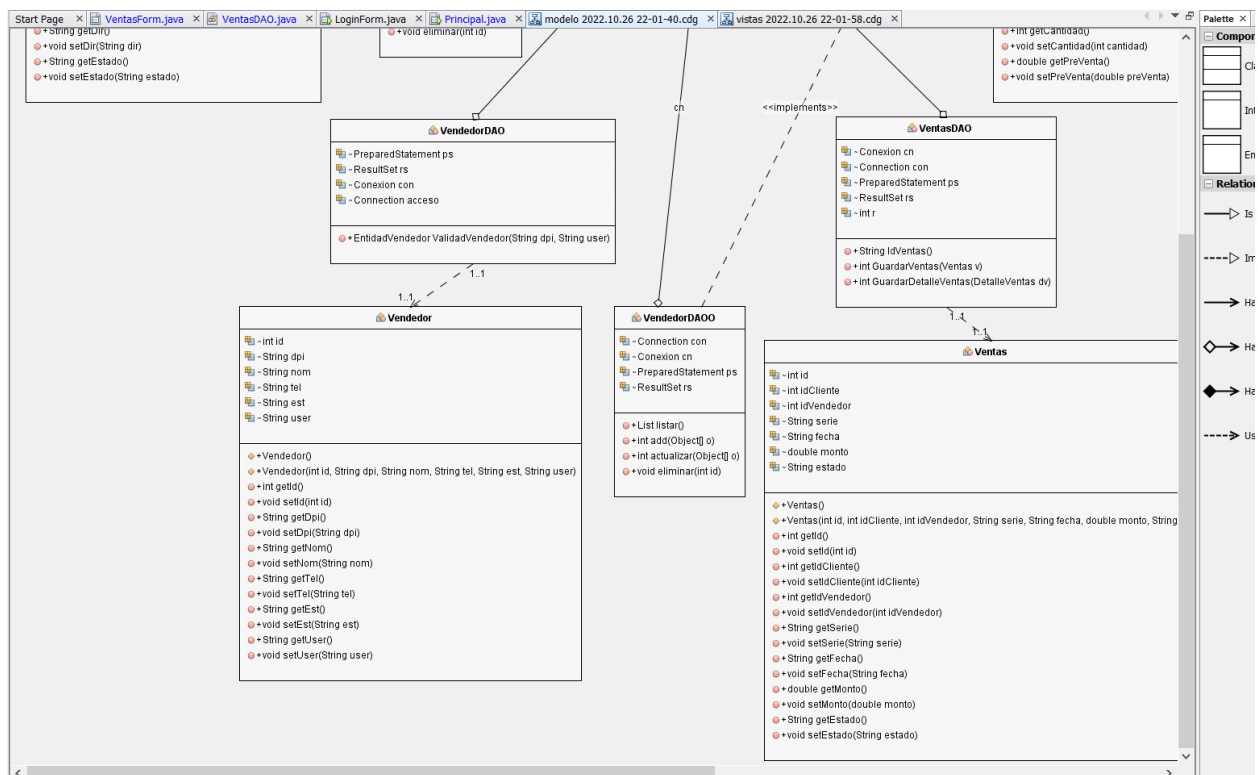
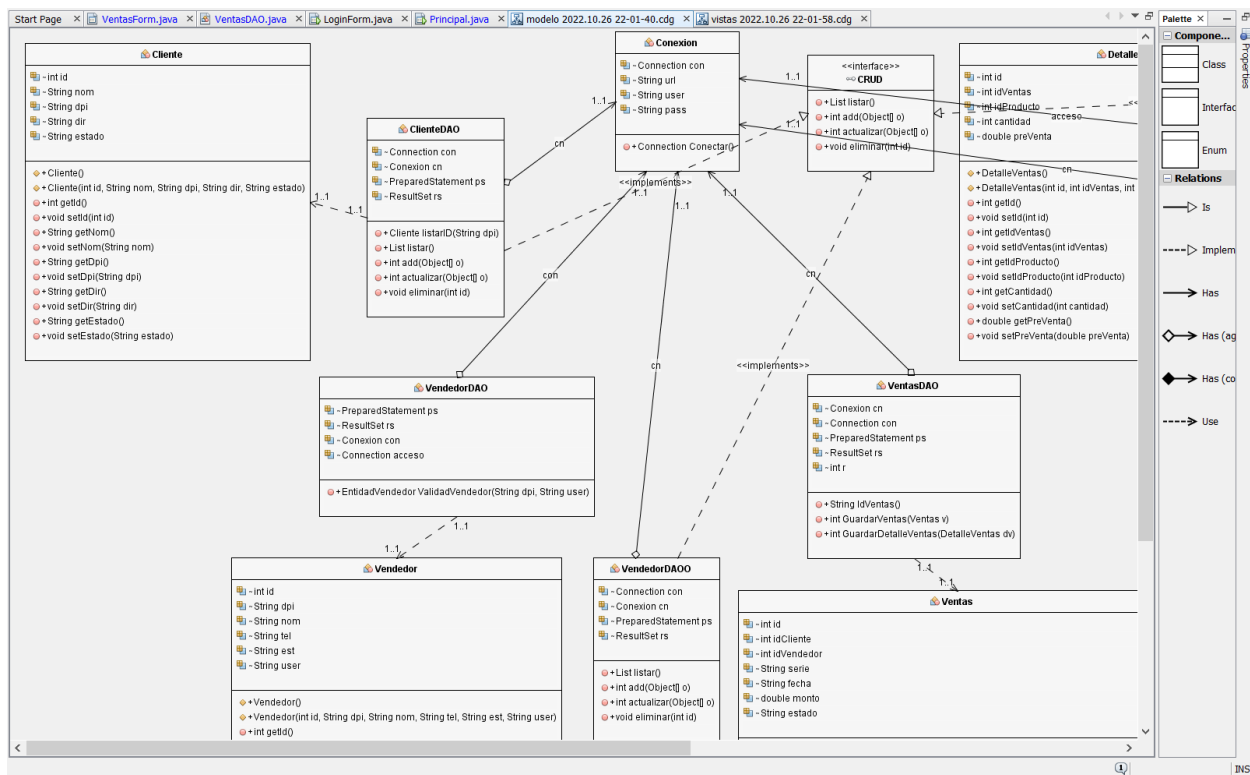
El 4 de junio de 2018, Microsoft compró GitHub. Al inicio, el cambio de propietario generó preocupaciones y la salida de algunos proyectos de este sitio; sin embargo, no fueron representativos. GitHub continúa siendo la plataforma más importante de colaboración para proyectos de código abierto.

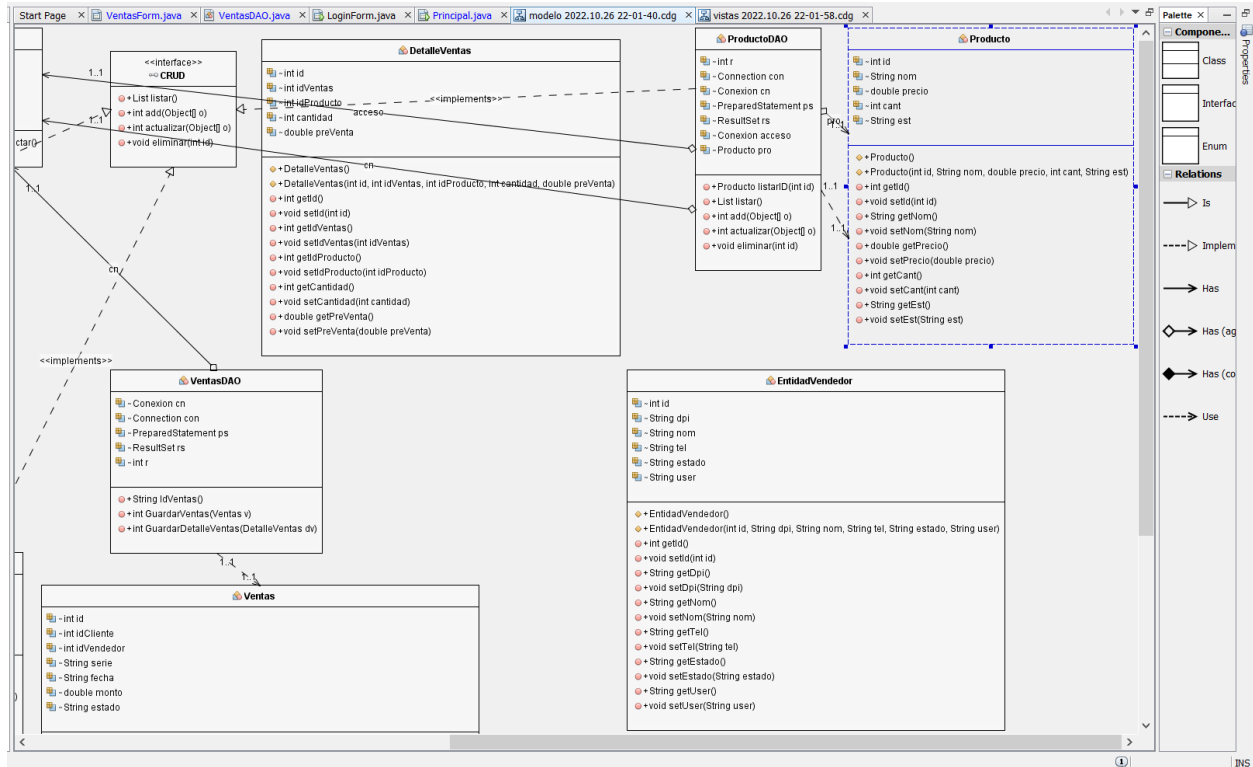
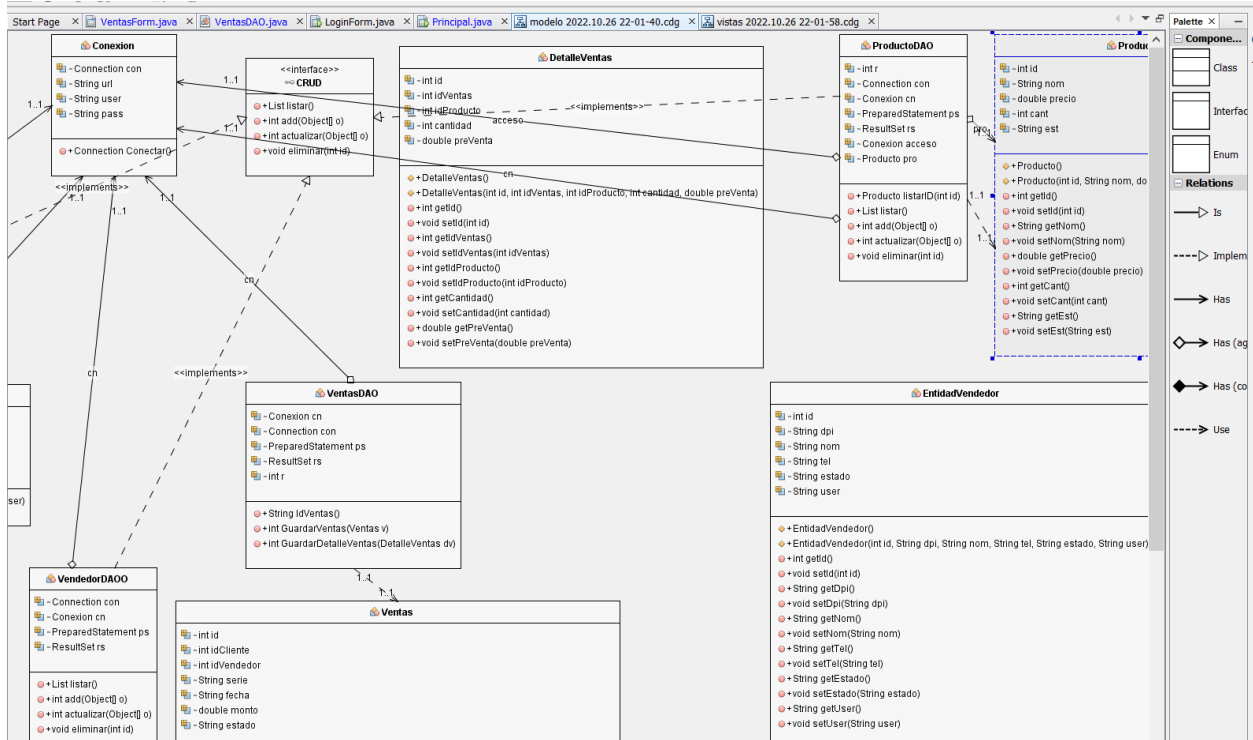
Los repositorios son, como su propio nombre indica, lugares virtuales alojados en la nube en donde los usuarios pueden almacenar cualquier tipo de archivo. Suelen usarse para guardar archivos de texto que representan código en lenguajes de programación, los cuales a su vez conforman un programa o aplicación. Este es el servicio principal que ofrece GitHub: repositorios en donde los usuarios pueden almacenar el código de sus aplicaciones, ya sea de forma pública, privada o a través de una organización.

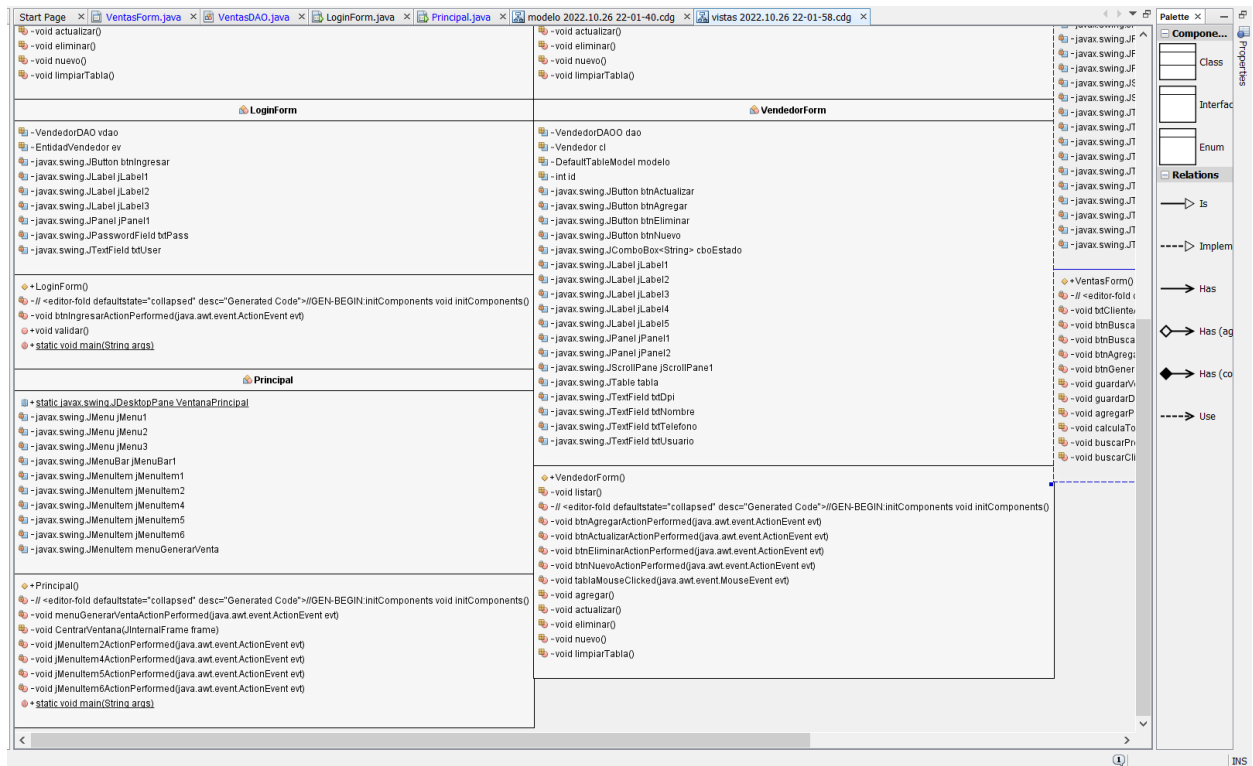
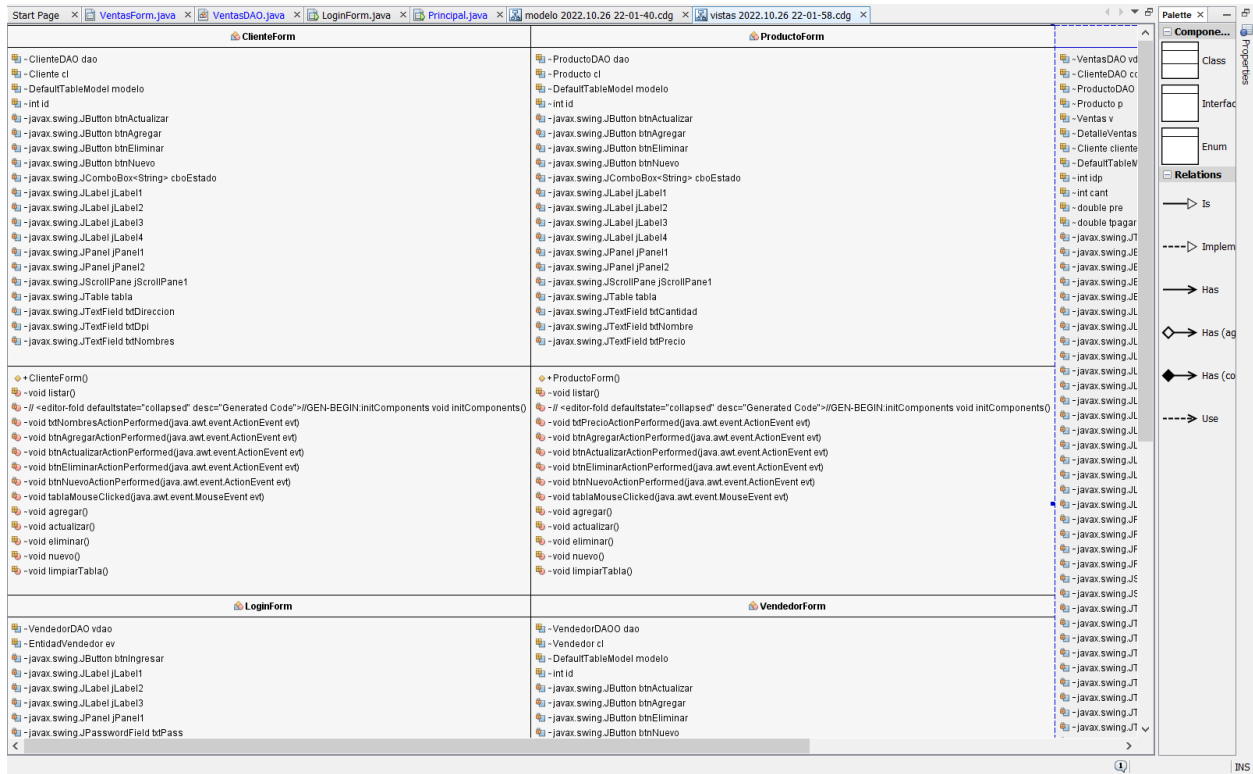
Para ser identificados, los repositorios muestran información relevante, como la descripción del proyecto, los lenguajes de programación usados o las etiquetas de búsqueda. Un aspecto muy importante de la información de los repositorios es el archivo README, el cual los desarrolladores pueden crear ya sea en formato Markdown o en formato de texto simple para describir su proyecto a fondo. Por lo general, este tipo de archivo suele contener una descripción, capturas de pantalla del programa, instrucciones de instalación y la declaración de la licencia.

En la pestaña de configuración, los desarrolladores pueden cambiar aspectos generales del repositorio, así como aspectos más avanzados. Entre los aspectos más avanzados se incluye el acceso al repositorio agregar colaboradores y determinar quiénes pueden interactuar con él, la automatización del y gracias al repositorio las ramas y sus reglas. GitHub Actions, webhooks, los ambientes y GitHub Pages, la seguridad del repositorio la seguridad y el análisis del código mediante alertas de bots y actions, las llaves de despliegue y los secretos cifrados y por último, las integraciones, que incluyen el uso de las aplicaciones instaladas en GitHub y las notificaciones por correo electrónico.

Diagrama de Clases







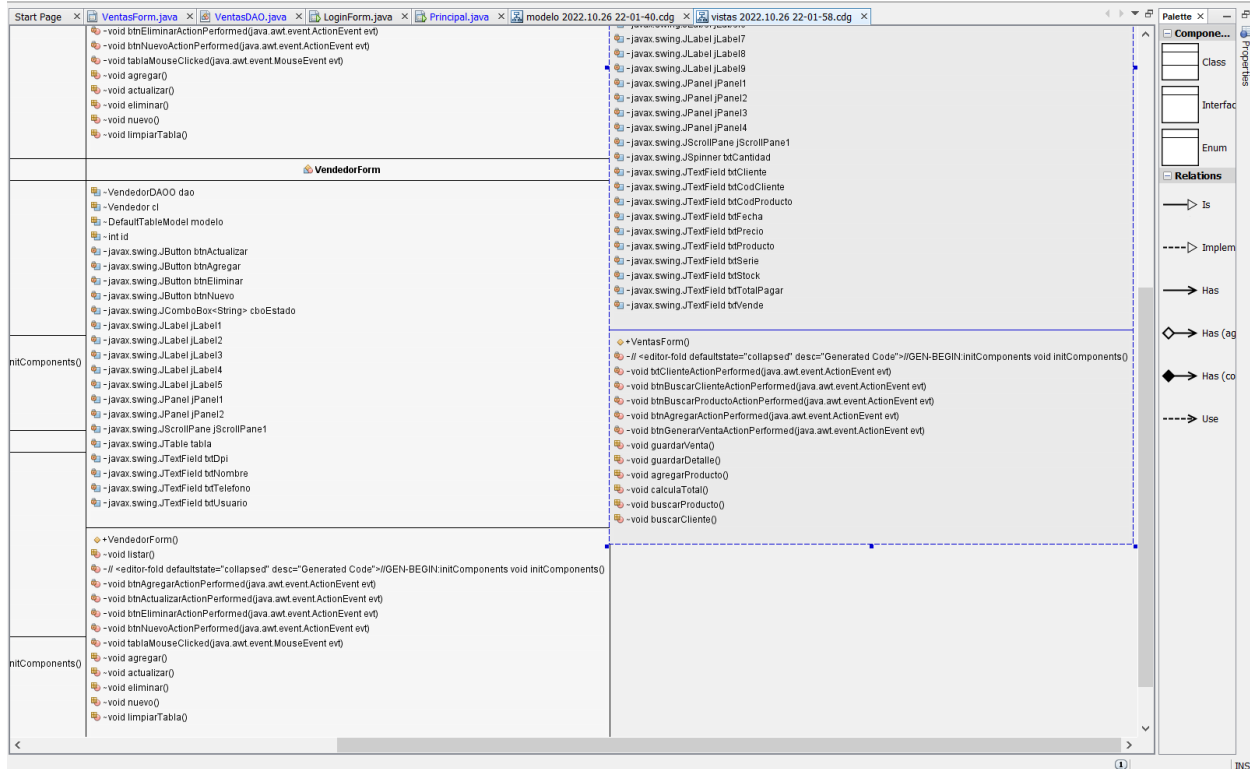
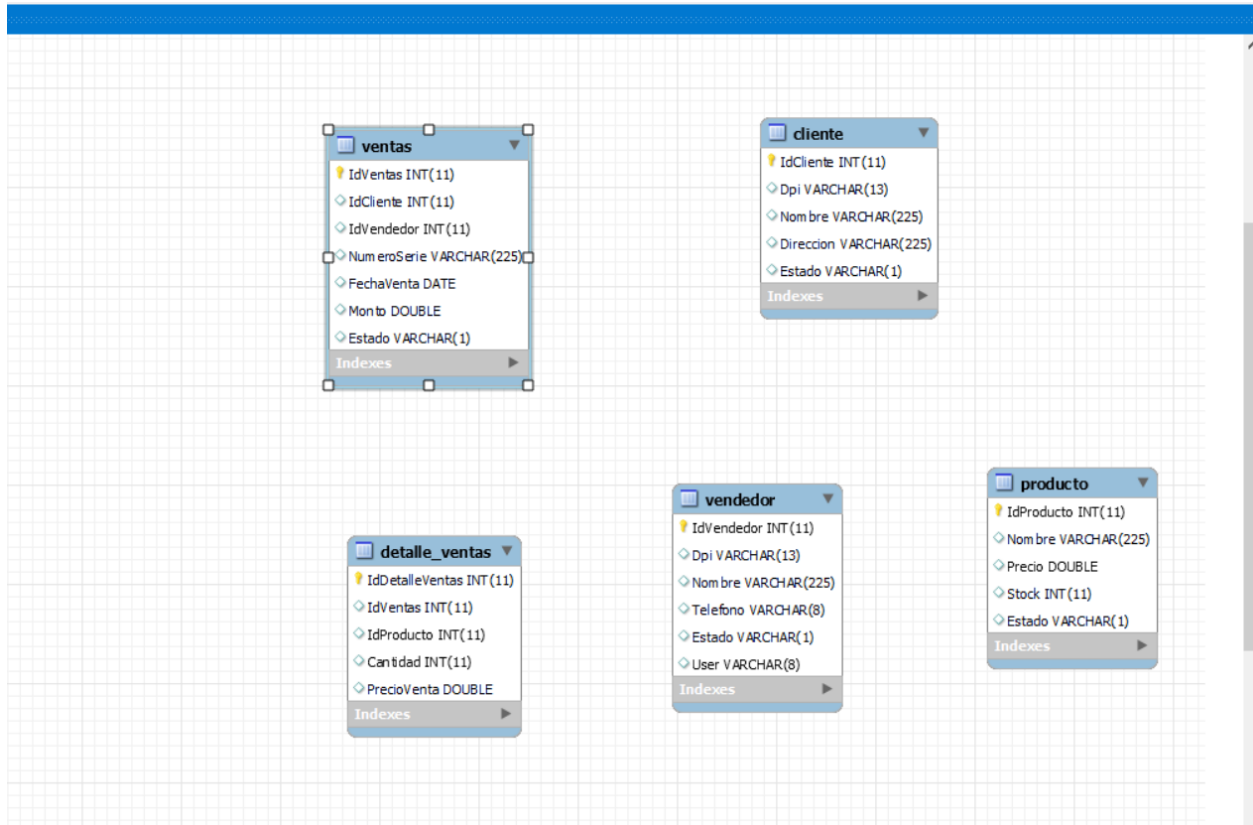
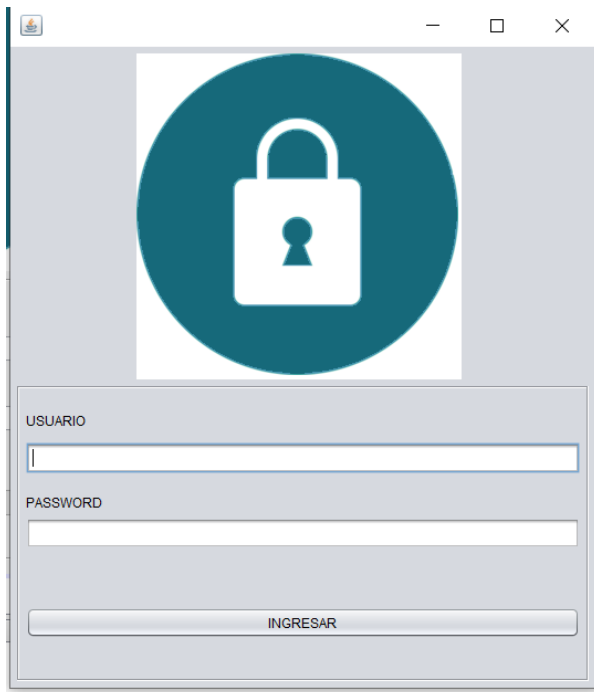


Diagrama de Base de Datos

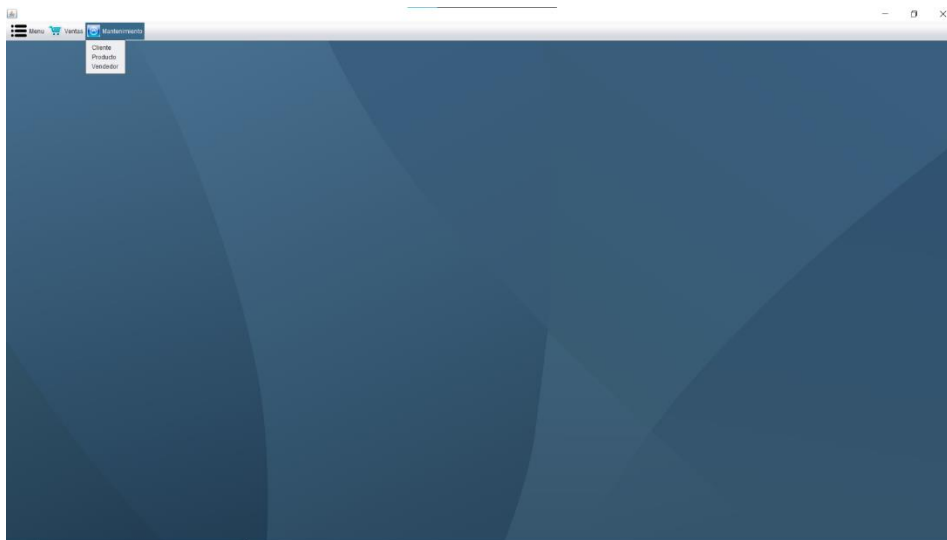


Manual de Usuario



VENTANA LOGIN

Al iniciar el programa, lo primero que aparecerá es la ventana de **“LOGIN”**, es la ventana de registro para poder ingresar al sistema de ventas, para acceder se debe de ingresar los respectivos datos que se pide, en este caso un nombre de usuario y un password o contraseña, para ello debe registrarse primero en la base de datos para que le permita ingresar, caso contrario no le permitirá ingresar.



VENTANA PRINCIPAL

En la ventana principal, contiene diferentes opciones que el usuario puede acceder a ello, los siguientes apartados son: menú, ventas, y mantenimiento.

Estas 3 opciones contienen diferentes funcionalidades para poder tener una mejor administración de una empresa de ventas.

En la opción de mantenimiento es donde se pueden registrar clientes, productos y proveedores.

Modulo Cliente

DPI: 321

NOMBRES: Francisco Sales

DIRECCION: El Valle

ESTADO: 1

ID	DPI	NOMBRES	DIRECCION	ESTADO
8	321	Francisco Sales	El Valle	1

MODULO CLIENTE

En el módulo cliente, es una opción donde se registran los clientes.

Para poder agregar un cliente se debe presionar el botón **“NUEVO”**, luego de eso se debe llenar los respectivos datos que se piden (dpi, nombres, dirección, estado (1 = activo, 0 = no activo)), luego de haber llenado se debe dar click en el botón **“AGREGAR”**.

Para poder eliminar a un cliente se debe de seleccionar o presionar el cliente a eliminar y luego presionar el botón **“ELIMINAR”**.

La opción de actualizar no funciona, lo que se recomienda es eliminar al cliente y luego agregarlo con los datos actualizados

Modulo Productos

NOMBRE: Libra de Arroz

PRECIO: 2.5

CANTIDAD: 100

ESTADO: 1

Id	Nombre	Precio	Cantidad	Estado
7	Libra de Arroz	2.5	100	1

MODULO PRODUCTOS

En el módulo producto, es la opción donde se registran los productos.

Para poder registrar un producto se debe presionar el botón **“NUEVO”**, luego de eso se debe llenar los respectivos datos que se piden (nombre del producto, precio, cantidad, estado (1 = activo, 0 = no activo)), luego de haber llenado se debe dar click en el botón **“AGREGAR”**.

Para poder eliminar a un producto se debe de seleccionar o presionar el producto a eliminar y luego presionar el botón **“ELIMINAR”**.

La opción de actualizar no funciona, lo que se recomienda es eliminar el producto y luego agregarlo con los datos actualizados

Modulo Vendedor

DPI: 12345678 AGREGAR

NOMBRE: Empleado 001 ACTUALIZAR

TELEFONO: 46722187 ELIMINAR

ESTADO: 1 NUEVO

USUARIO: empl001

ID	DPI	NOMBRE	TELEFONO	ESTADO	USUARIO
1	12345678	Empleado 0...	46722187	1	empl001


MODULO VENDEDOR

En el módulo vendedor, es la opción donde se registran los vendedores.

Para poder registrar un vendedor se debe presionar el botón **“NUEVO”**, luego de eso se debe llenar los respectivos datos que se piden (dpi, precio del empleado, teléfono, estado (1 = activo, 0 = no activo) y nombre de usuario), luego de haber llenado se debe dar click en el botón **“AGREGAR”**.

Para poder eliminar a un vendedor se debe de seleccionar el vendedor a eliminar y luego presionar el botón **“ELIMINAR”**.

La opción de actualizar no funciona, lo que se recomienda es eliminar el producto y luego agregarlo con los datos actualizados



Sistema de Ventas

PUNTO DE VENTA "El Abarrotero"

Venta de Abarros

Tel: 24280000

COD: CLIENTE: 321

BUSCAR

CLIENTE: Francisco Sales

COD: PRODUCTO: 7

BUSCAR

PRODUCTO: Libra de Arroz

PRECIO: 2.5

AGREGAR

STOCK: 100

CANTIDAD: 1

2022-9-27

GENERAR VENTA

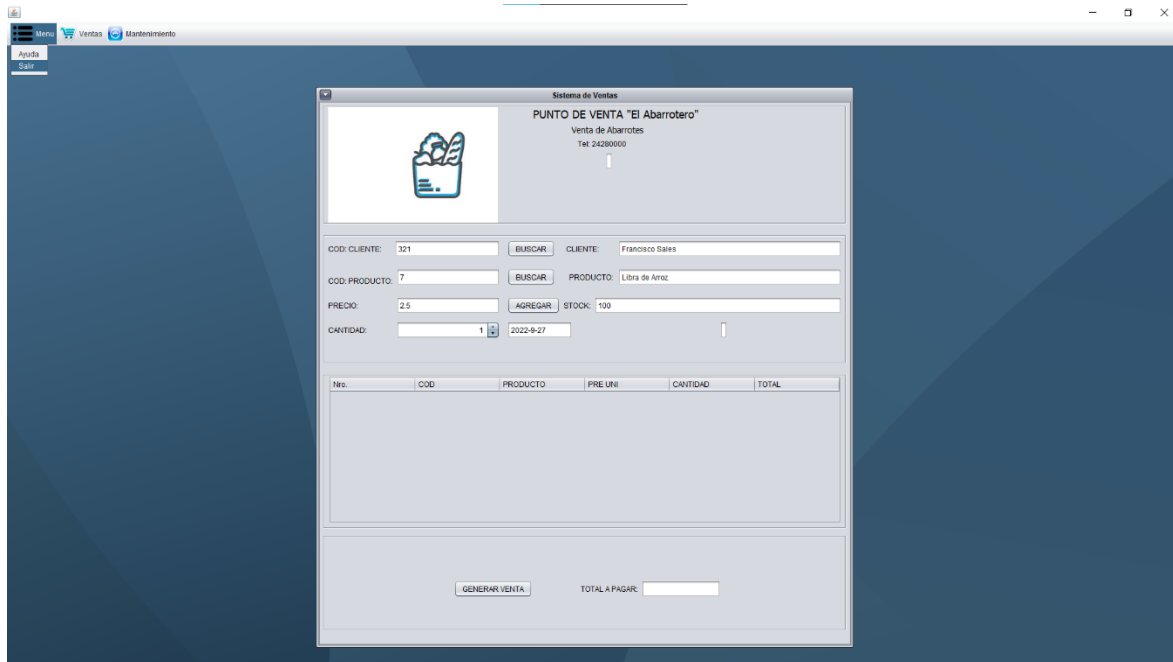
TOTAL A PAGAR:

SISTEMA DE VENTAS

En el sistema de ventas, es la interfaz donde se registran las ventas.

Para poder registrar una venta se deben de llenar los datos que se le pide.

- Primero, ingresar el código del cliente, luego presionar el botón **"BUSCAR"**, automáticamente aparecerá el nombre del cliente en la otra casilla, de igual manera lo puede ingresar de manera manual.
- Segundo, ingresar el código del producto, luego presionar el botón **"BUSCAR"**, automáticamente se llenarán los datos del producto en la casilla de precio, stock y nombre del producto.
- Por último se debe de ingresar la cantidad del producto a vender, luego de todo eso para poder generar la venta se debe de dar click o presionar al botón de **"AGREGAR"**, de esa manera se podrá generar la venta indicando el precio total a pagar.



VENTANA PRINCIPAL

Una de las opciones que aparece en la ventana principal, es la opción de menú, en menú aparece la opción de ayuda y salir.

La opción de ayuda actualmente no cuenta con funcionalidad, para salir del sistema de ventas o el programa solo debe de presionar la opción de **“Salir”**.

Conclusiones

Se concluyó que existía la necesidad de desarrollar un sistema de ventas bajo entorno web con el propósito de mejorar notablemente la manera en la que se llevan las actividades antes de ventas del consumo diario, con el fin de brindar una herramienta automatizada.

La conclusión de que el sistema de información bajo ambiente automatizado para la venta de productos es electrónicos para ventas de productos de gran cantidad y que ayudará a mejorar las actividades de ventas con un mayor eficiencia de trabajo y rapidez por lo cual genera un mayor orden.

Finalmente se logró demostrar mediante las pruebas pertinentes la correcta ejecución del sistema de información que su función es un es llevar un mejor control y orden de la venta de artículos, considerando que el proyecto es operativo, ya que en los aspectos generales cumple con los indicadores propuestos con el uso de POO.