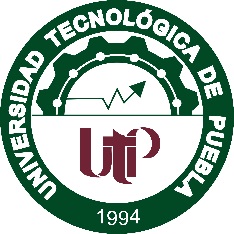
**Tecnologías de la información y Comunicación**

Proyecto Final

Punto de venta con conexión a una base de datos remota

Integrantes:

Abraham Loranca Rosas

Materia:

Administración De Base De Datos

Profesor:

Arturo rojas López

Cuatrimestre:

Enero-Abril 2016

Grado y Grupo:

4.-A

Contenido

[**Problemática** 3](#_Toc448825172)

[**Detalles del servidor** 4](#_Toc448825173)

[Esquema de la base de datos 4](#_Toc448825174)

[Código de creación base de datos 5](#_Toc448825175)

[Capturas de pantalla Formulario 13](#_Toc448825176)

[Código formulario 15](#_Toc448825177)

[Código Clase 26](#_Toc448825178)

[Capturas del programa corriendo desde el .exe 37](#_Toc448825179)

# **Problemática**

Cristal helado

La empresa Cristal helado ubicada en el municipio de Chignahuapan Puebla ha diseñado y fabricado esferas navideñas por más de 25 años, su dueño Rafael Noel Pérez Olmos inicio en la elaboración del producto junto con sus dos hijos (Pánfilo y Ximena) y su esposa Victoria, en la actualidad la empresa está constituida por 5 empleados coordinados por el Sr. Noel. El éxito comercial radica en la especialización del tipo de producto que elabora: esfera dorada, plateada y roja, en dos tamaños: 3 y 5 centímetros de diámetro. La venta la realiza por unidad o por caja, la cual puede ser de 6, 12 o 24 unidades.

El costo de producción está determinado entre otras cosas por el precio de los insumos, tales como: el litro de pintura, el kilo de cartón para el embalaje según las unidades a empacar y el kilo del gancho usado para colgar la esfera que es el mismo para los dos tamaños.

Un litro de pintura rinde para 300 unidades de 3 centímetros de diámetro, mientras para las de 5 centímetros rinde para 265 unidades. Un kilo de cartón para almacenar 6 unidades rinde para armar 30 cajas, para almacenar 12 unidades el kilo rinde para armar 20 cajas y finalmente para almacenar 24 unidades rinde para armar 15 cajas. Un kilo de ganchos alcanza para 700 esferas. Una vez que obtiene los precios de producción calcula el costo de fabricar una esfera y entonces para la venta aumenta un 30 % el costo de producción y obtiene el precio de esfera al público.

Por ejemplo, si el litro de pintura cuesta $200, y el kilo de gancho cuesta $250, para producir una esfera dorada de 5 centímetros requiere de  $1.11 (0.75-pintura + 0.36-gancho), entonces al aumentarle $0.33, nos genera el precio de venta de $1.44. Un segundo ejemplo para obtener el precio de venta de una caja de esfera roja de 3 centímetros de diámetro de 12 unidades es el siguiente: si el litro de pintura cuesta $250, el kilo de cartón $80 y el kilo de gancho cuesta $240, para producir la caja requiere de $18.04 (9.96-pintura + 4-cartón + 4.08-gancho), entonces al aumentarle $5.41, nos genera el precio de venta $23.45.

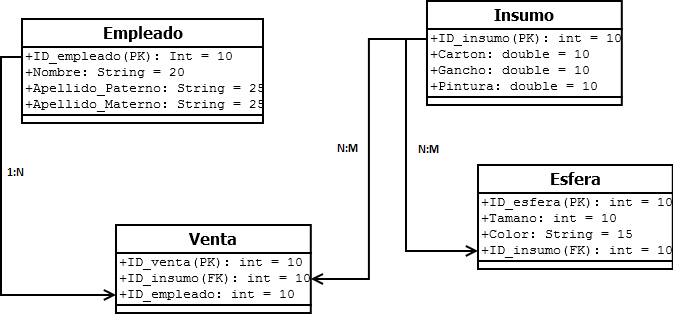
Crear un sistema que permita al Sr. Noel realizar la venta de su producto al público de una forma rápida considerando la compra de los insumos que permiten calcular los precios de la esfera. El software elaborará una factura al cliente que lo requiera, agregando el IVA respectivo al precio de venta o bien una nota que describa la compra realizada. La aplicación llevará un control de las ventas realizadas para poder realizar consultas de aquellas que solicitaron factura o nota y cuando se requiera ver el detalle de la venta respectiva, es decir, poder visualizar las ventas y cuando se seleccione una en particular poder ver el detalle de la misma (fecha, vendedor, producto(s), cantidades de la venta, etc.)

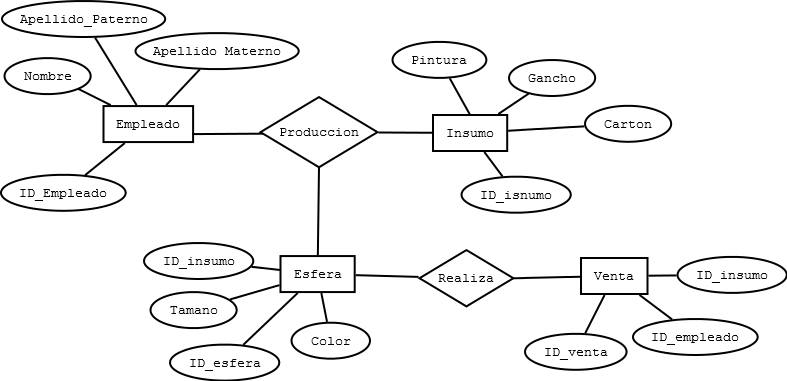
Utilizar una base de datos para gestionar la información de la empresa, conectar la aplicación a la base de datos y ofrecer las consultas de ventas indicadas además de la funcionalidad de la venta misma.

# **Detalles del servidor**

Se ha determinado utilizar un servidor externo para que gestione la base de datos, el servidor seleccionado se encuentra en el host db4free.net. Para hacer uso de dicha base de datos, se utilizará un enlace desde la aplicación para que realice la conexión remota a la base de datos de manera automática

# Esquema de la base de datos





# Código de creación base de datos

-- phpMyAdmin SQL Dump

-- version 4.6.0

-- http://www.phpmyadmin.net

--

-- Servidor: 127.0.0.1:3306

-- Tiempo de generación: 18-04-2016 a las 06:43:17

-- Versión del servidor: 5.7.12

-- Versión de PHP: 5.5.9-1ubuntu4.14

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

SET time\_zone = "+00:00";

--

-- Base de datos: `esferas123`

--

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `factura`

--

CREATE TABLE `factura` (

`id\_factura` int(11) NOT NULL,

`tipo\_compra` varchar(30) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`fecha\_venta` date NOT NULL,

`Total` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Volcado de datos para la tabla `factura`

--

INSERT INTO `factura` (`id\_factura`, `tipo\_compra`, `fecha\_venta`, `Total`) VALUES

(5, 'factura', '2016-04-14', 1296),

(6, 'factura', '2016-04-14', 1367),

(7, 'factura', '2016-04-14', 865),

(8, 'factura', '2016-04-14', 925),

(9, 'factura', '2016-04-14', 439),

(10, 'nota', '2016-04-14', 325),

(11, 'factura', '2016-04-15', 1562),

(12, 'factura', '2016-04-15', 304);

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `Insumos`

--

CREATE TABLE `Insumos` (

`id\_insumo` int(11) NOT NULL,

`precio\_carton` decimal(10,0) NOT NULL,

`precio\_gancho` decimal(10,0) NOT NULL,

`precio\_pintura` decimal(10,0) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Volcado de datos para la tabla `Insumos`

--

INSERT INTO `Insumos` (`id\_insumo`, `precio\_carton`, `precio\_gancho`, `precio\_pintura`) VALUES

(49, 200, 200, 200),

(50, 20, 200, 200),

(51, 20, 200, 200),

(52, 202, 20, 20),

(53, 80, 250, 200),

(54, 80, 250, 200),

(55, 80, 250, 200),

(56, 80, 250, 200),

(57, 80, 250, 200),

(58, 200, 200, 200),

(59, 200, 200, 200),

(60, 200, 200, 200),

(61, 200, 200, 200),

(62, 23, 50, 100),

(63, 250, 80, 432),

(64, 250, 80, 432);

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `subtotal`

--

CREATE TABLE `subtotal` (

`sub\_total` int(11) NOT NULL,

`id\_factura` int(11) NOT NULL,

`vendedor` varchar(30) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

`producto` varchar(30) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

`color` varchar(20) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

`unidades` int(11) DEFAULT NULL,

`cantidad` int(11) DEFAULT NULL,

`subtotal` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Volcado de datos para la tabla `subtotal`

--

INSERT INTO `subtotal` (`sub\_total`, `id\_factura`, `vendedor`, `producto`, `color`, `unidades`, `cantidad`, `subtotal`) VALUES

(1, 5, 'Rafel Noel', 'Venta por caja', 'plateada', 12, 10, 405),

(2, 5, 'Rafel Noel', 'Venta por caja', 'plateada', 12, 22, 891),

(3, 6, 'Rafel Noel', 'Venta por caja', 'dorada', 6, 10, 230),

(4, 6, 'Rafel Noel', 'Venta por caja', 'plateada', 12, 10, 435),

(5, 6, 'Rafel Noel', 'Venta por caja', 'roja', 24, 10, 702),

(9, 7, 'Rafel Noel', 'Venta por caja', 'plateada', 12, 6, 261),

(10, 7, 'Rafel Noel', 'Venta por caja', 'plateada', 6, 6, 147),

(11, 7, 'Rafel Noel', 'Venta por caja', 'plateada', 24, 6, 457),

(13, 8, 'Rafel Noel', 'Venta por caja', '', 0, 1, 23),

(14, 8, 'Rafel Noel', 'Venta por unidad', 'plateada', 12, 10, 850),

(15, 8, 'Rafel Noel', 'Venta por caja', 'dorada', 0, 21, 52),

(16, 9, 'Rafel Noel', 'Venta por unidad', 'dorada', 0, 15, 34),

(17, 9, 'Rafel Noel', 'Venta por caja', 'plateada', 12, 10, 405),

(18, 10, 'Rafel Noel', 'Venta por unidad', 'dorada', 0, 15, 34),

(19, 10, 'Rafel Noel', 'Venta por caja', 'roja', 6, 10, 230),

(20, 10, 'Rafel Noel', 'Venta por unidad', 'plateada', 0, 30, 61),

(21, 11, 'Rafel Noel', 'Venta por unidad', 'plateada', 0, 20, 40),

(22, 11, 'Rafel Noel', 'Venta por caja', 'plateada', 24, 20, 1522),

(23, 12, 'Rafel Noel', 'Venta por caja', 'roja', 12, 2, 87),

(24, 12, 'Rafel Noel', 'Venta por caja', 'dorada', 12, 5, 217);

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `tiquet`

--

CREATE TABLE `tiquet` (

`id\_tiquet` int(11) NOT NULL,

`vendedor` varchar(30) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`descripcion` varchar(30) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`fecha\_venta` date NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `Trabajador`

--

CREATE TABLE `Trabajador` (

`id\_trabajador` int(11) NOT NULL,

`nombre` varchar(30) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`apellido\_paterno` varchar(30) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`apellido\_materno` varchar(30) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`edad` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Volcado de datos para la tabla `Trabajador`

--

INSERT INTO `Trabajador` (`id\_trabajador`, `nombre`, `apellido\_paterno`, `apellido\_materno`, `edad`) VALUES

(9, 'Rafel Noel', 'Perez', 'Olmos', 33),

(10, 'Panfilo', 'Perez', 'Olmos', 19),

(11, 'Ximena', 'Perez', 'Olmos', 21),

(12, 'Luis', 'Hernandez', 'Lopez', 21),

(13, 'Jesus', 'Hernandez', 'Gonzales', 25);

--

-- Índices para tablas volcadas

--

--

-- Indices de la tabla `factura`

--

ALTER TABLE `factura`

ADD PRIMARY KEY (`id\_factura`);

--

-- Indices de la tabla `Insumos`

--

ALTER TABLE `Insumos`

ADD PRIMARY KEY (`id\_insumo`);

--

-- Indices de la tabla `subtotal`

--

ALTER TABLE `subtotal`

ADD PRIMARY KEY (`sub\_total`);

--

-- Indices de la tabla `tiquet`

--

ALTER TABLE `tiquet`

ADD PRIMARY KEY (`id\_tiquet`);

--

-- Indices de la tabla `Trabajador`

--

ALTER TABLE `Trabajador`

ADD PRIMARY KEY (`id\_trabajador`);

--

-- AUTO\_INCREMENT de las tablas volcadas

--

--

-- AUTO\_INCREMENT de la tabla `factura`

--

ALTER TABLE `factura`

MODIFY `id\_factura` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=13;

--

-- AUTO\_INCREMENT de la tabla `Insumos`

--

ALTER TABLE `Insumos`

MODIFY `id\_insumo` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=65;

--

-- AUTO\_INCREMENT de la tabla `subtotal`

--

ALTER TABLE `subtotal`

MODIFY `sub\_total` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=25;

--

-- AUTO\_INCREMENT de la tabla `tiquet`

--

ALTER TABLE `tiquet`

MODIFY `id\_tiquet` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `Trabajador`

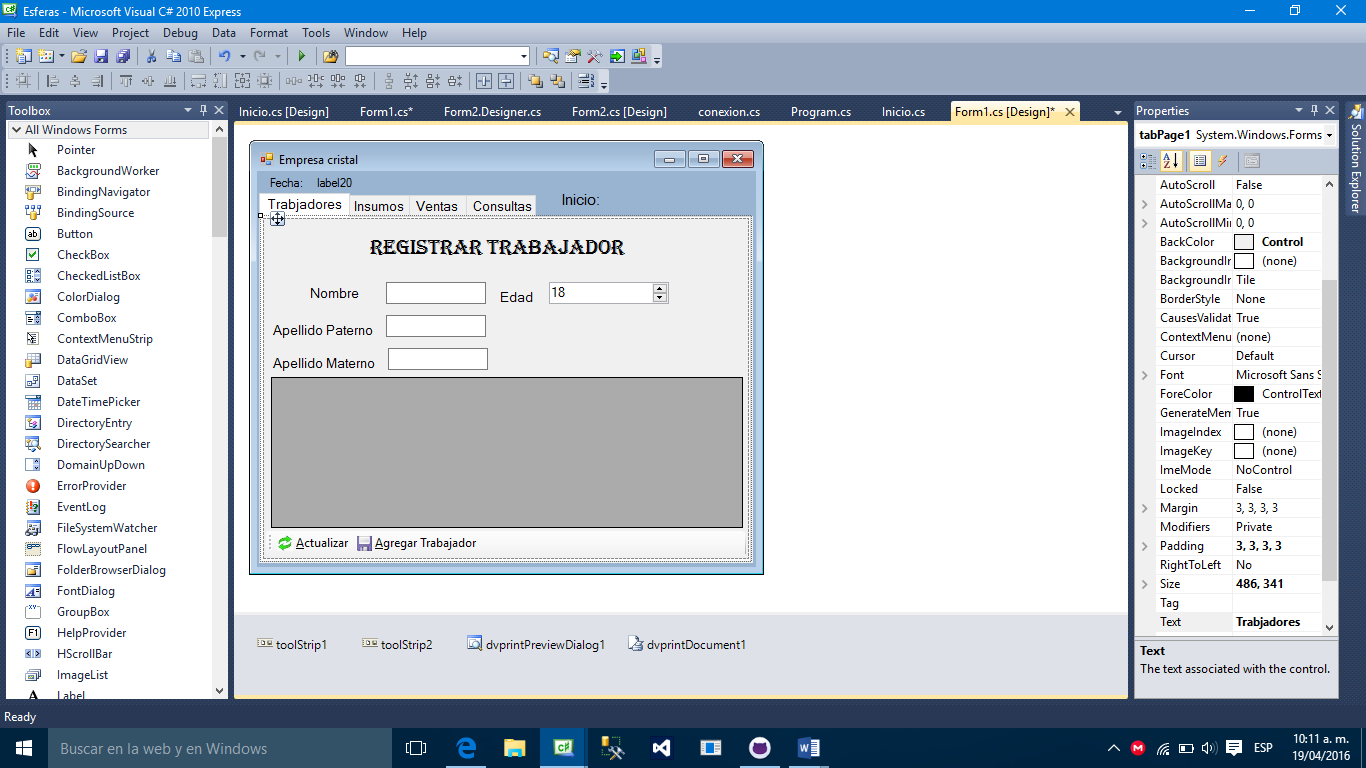
--

ALTER TABLE `Trabajador`

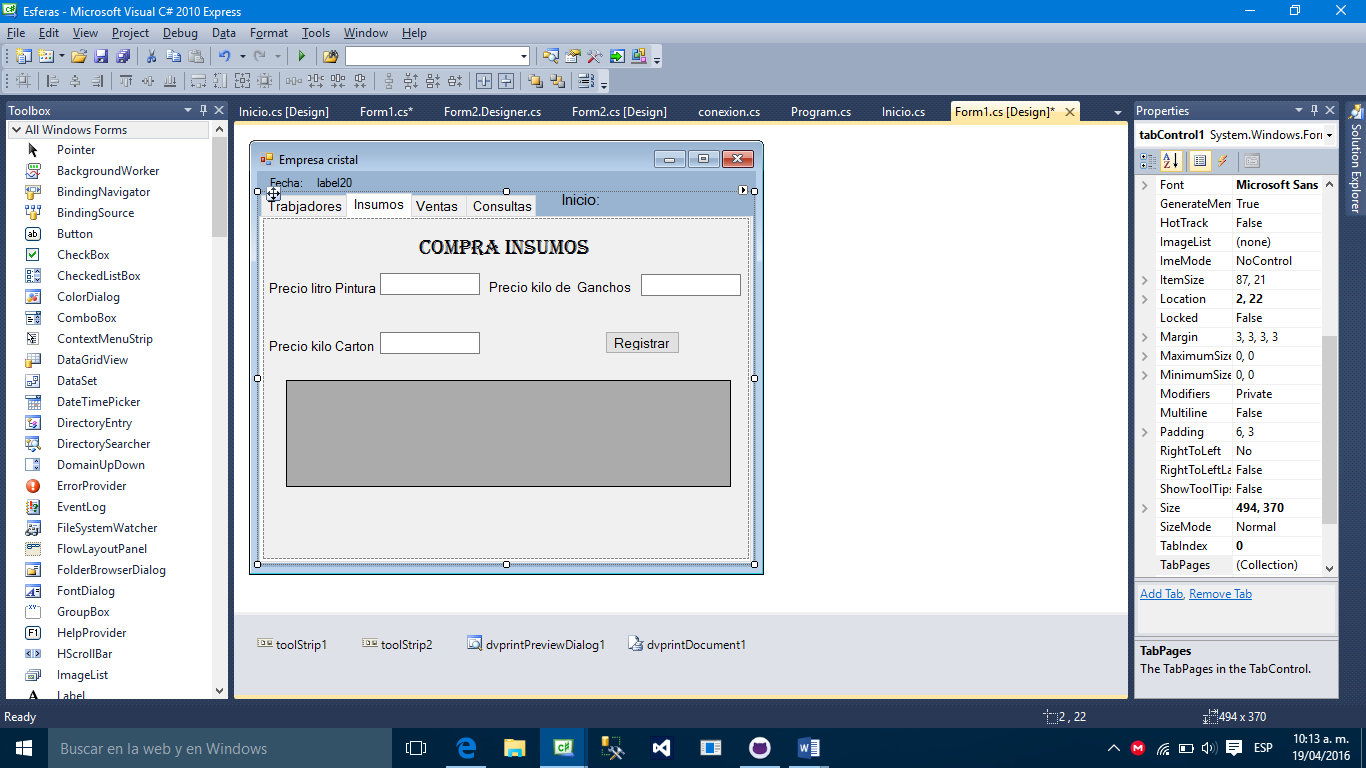
MODIFY `id\_trabajador` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=14;

# Capturas de pantalla Formulario

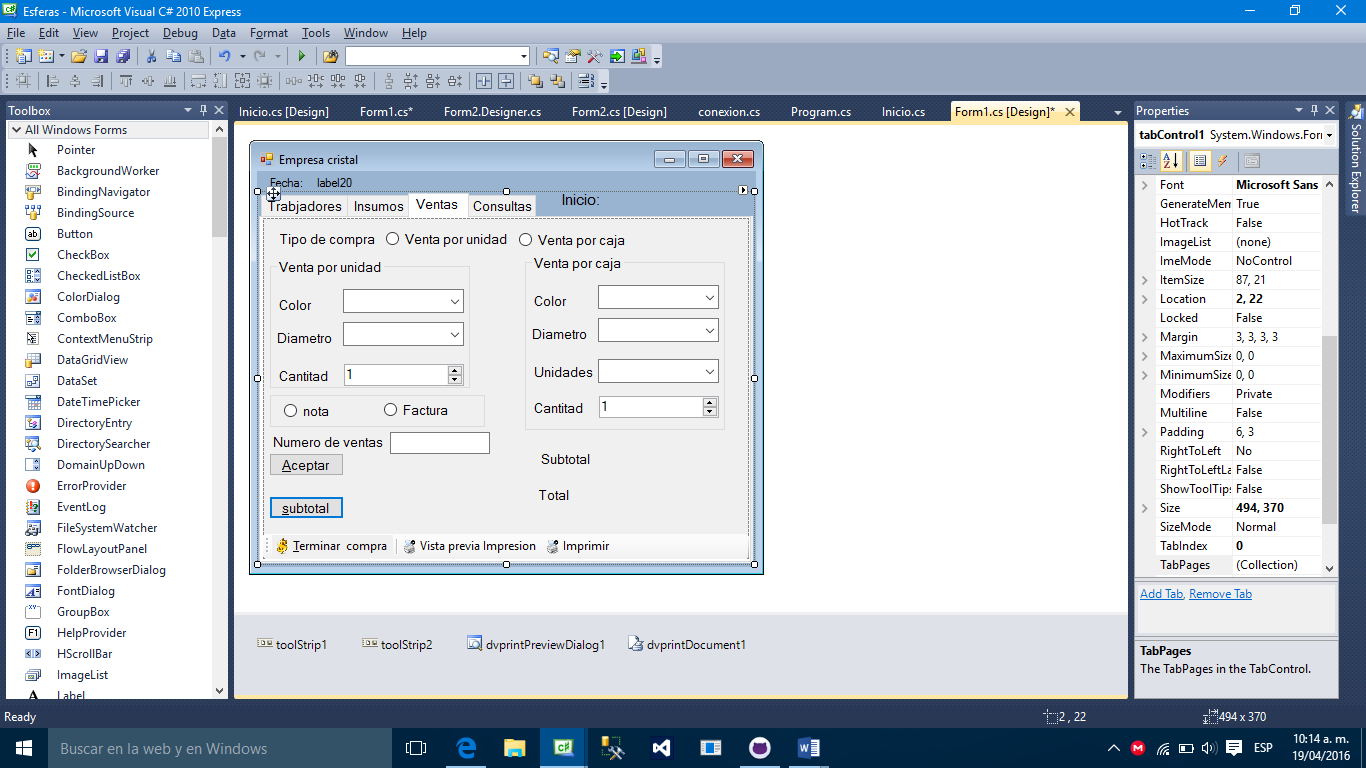
Esta ventana es de trabajadores donde se registrara los trabajadores



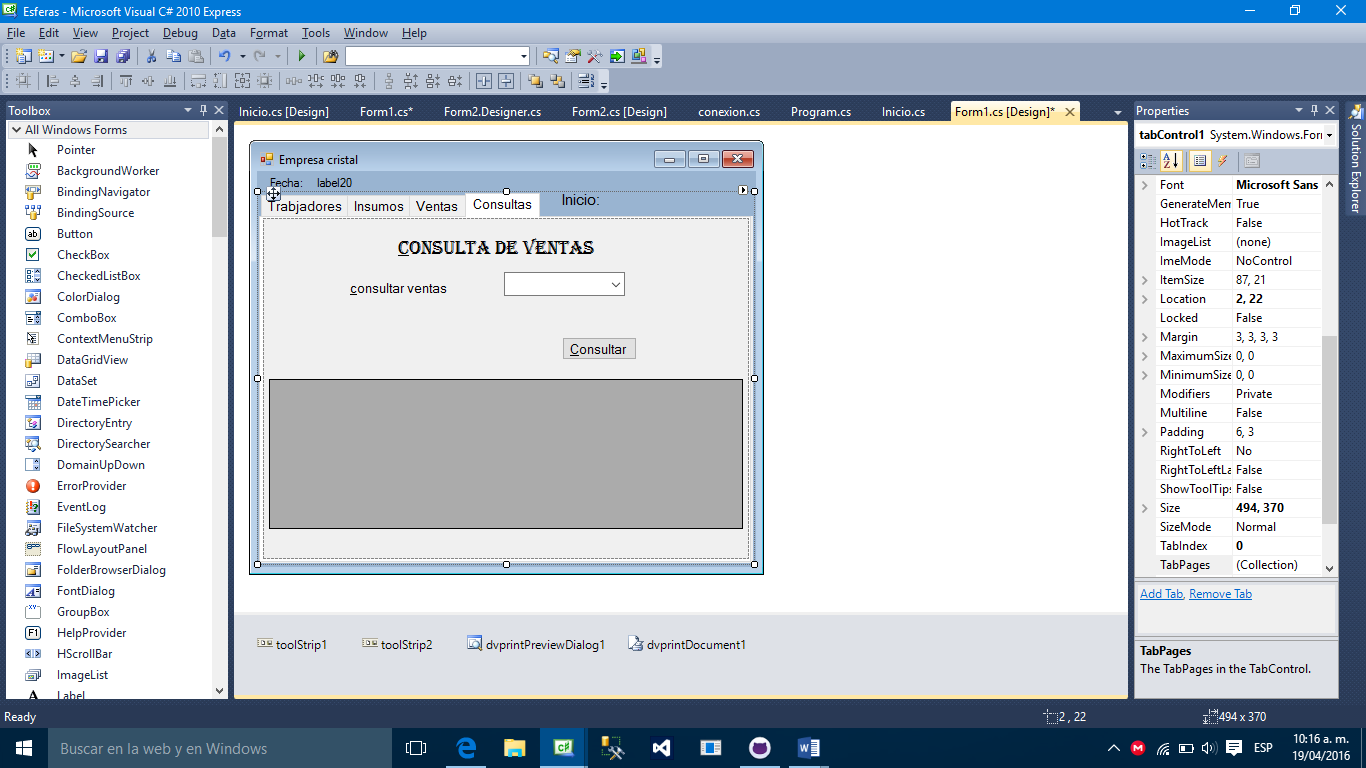
Ventana Insumos donde se registran los precios de los insumos



Ventana ventas el trabajador seleccionara cuantas ventas desea realizar, desde esta ventana selecciona los productos a vender



Ventana consultas se podrá ver qué ventas se realizaron dependiendo si fue por factura o nota simple



# Código formulario

namespace Esferas

{

public partial class Form1 : Form

{

int vent;

Conexion nombreInstancia = new Conexion();

Totales objeto = new Totales();

double precioc, preciog, preciolip;

subtotal total;

int countRows = 0;

string preciocarton = "Precio\_carton";

string preciogancho= "precio\_ganchos";

string preciopintura = "precio\_pintura";

double gancho, pintura, carton;

insumos pCliente = new insumos();

public Form1()

{

InitializeComponent();

}

class Conexion

{

public string datos = @"server=db4free.net;database=esferas123;port=3306;Uid=cabresto5; pwd=cabresto5;";

public MySqlConnection conexion;

public void abrir()

{

try

{

conexion = new MySqlConnection(datos);

conexion.Open();

MessageBox.Show("conexion realizada exitosamente");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

public void cerrar()

{

try

{

conexion = new MySqlConnection(datos);

conexion.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

private void button2\_Click(object sender, EventArgs e)

{

nombreInstancia.abrir();

}

private void label8\_Click(object sender, EventArgs e)

{

}

private void rdbUnidad\_CheckedChanged(object sender, EventArgs e)

{

groupBox2.Enabled = false;

groupBox1.Enabled = true;

}

private void Form1\_Load(object sender, EventArgs e)

{

rdbUnidad.Checked = true;

groupBox1.Enabled = true;

groupBox2.Enabled = false;

}

private void rdbCaja\_CheckedChanged(object sender, EventArgs e)

{

groupBox2.Enabled = true;

groupBox1.Enabled = false;

}

//boton de insumos

private void button3\_Click(object sender, EventArgs e)

{

preciocarton = txtkcarton.Text;

preciogancho = txtkganchos.Text;

preciopintura = txtlitrop.Text;

//mandar a la base de datos los insumos

pCliente.precio\_carton = txtkcarton.Text;

pCliente.precio\_ganchos = txtkganchos.Text;

pCliente.precio\_pintura = txtlitrop.Text;

int resultado = insumosdad.Agregar(pCliente);

if (resultado > 0)

{

MessageBox.Show("insumo Guardado Con Exito!!", "Guardado", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("No se pudo guardar el insumo", "Fallo!!", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

dataGridView2.DataSource = insumosdad.cosultainsu();

}

private void button4\_Click(object sender, EventArgs e)

{

}

private void Form1\_Load\_1(object sender, EventArgs e)

{

rdfactura.Checked = true;

dataGridView2.DataSource = insumosdad.cosultainsu();

dataGridView1.DataSource = ClientesDAL.Buscar();

label18.Enabled = false;

rdbUnidad.Checked = true;

rdbUnidad.Enabled = false;

rdbCaja.Enabled = false;

groupBox1.Enabled = false;

groupBox2.Enabled = false;

label20.Text = DateTime.Now.ToShortDateString();

countRows= dataGridView2.Rows.Count;

carton =Convert.ToInt32(dataGridView2[preciocarton, countRows -1].Value);

pintura = Convert.ToInt32(dataGridView2[preciopintura, countRows - 1].Value);

gancho = Convert.ToInt32(dataGridView2[preciogancho, countRows - 1].Value);

pCliente.precio\_carton = carton.ToString();

pCliente.precio\_ganchos = gancho.ToString();

pCliente.precio\_pintura = pintura.ToString();

}

private void btnactualizar\_Click(object sender, EventArgs e)

{

}

private void btnactua\_Click(object sender, EventArgs e)

{

dataGridView1.DataSource = ClientesDAL.Buscar();

MessageBox.Show("informacion actualizada");

}

private void toolStripButton1\_Click(object sender, EventArgs e)

{

Cliente pCliente = new Cliente();

pCliente.Nombre = txtnombre.Text.Trim();

pCliente.Apellidop = txtapellidop.Text.Trim();

pCliente.apellidom = txtapellidom.Text.Trim();

pCliente.edad = numedad.Value.ToString();

int resultado = ClientesDAL.Agregar(pCliente);

if (resultado > 0)

{

MessageBox.Show("Cliente Guardado Con Exito!!", "Guardado", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("No se pudo guardar el cliente", "Fallo!!", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

private void dataGridView2\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void txttrabajdor\_TextChanged(object sender, EventArgs e)

{

}

private void rdbCaja\_CheckedChanged\_1(object sender, EventArgs e)

{

groupBox2.Enabled = true;

groupBox1.Enabled = false;

}

private void rdbUnidad\_CheckedChanged\_1(object sender, EventArgs e)

{

groupBox1.Enabled = true;

groupBox2.Enabled = false;

}

private void label20\_Click(object sender, EventArgs e)

{

}

private void toolStripButton2\_Click(object sender, EventArgs e)

{

DateTime dd = DateTime.Now;

/\*/this.vendedor = ven;

this.producto = product;

this.tipo\_compra = tipo\_com;

this.color = colo;

this.unidades = unidad;

this.cantidad = cant;

this.fechaventa = fecha;

this.Total = tot;/\*/

if (vent == Convert.ToUInt32(txtventas.Text))

{

button4.Enabled = true;

txtventas.Enabled = true;

if (precioc != 0 && preciog != 0 && preciolip != 0)

{

if (rdbCaja.Checked)

{

if (cmbcolorcaja.Text != "" && cmbcajaunidades.Text != "" && cmbdiamcaja.Text != "")

{

ventasfact pCliente = new ventasfact();

if (rdfactura.Checked)

{

pCliente.tipo\_compra = "factura";

}

else

{

pCliente.tipo\_compra = "nota";

}

pCliente.fechaventa = dd.ToString("yyyy/MM/dd");

label29.Text = total.sumasubtotal().ToString();

pCliente.Total = label29.Text;

int resultado = ventafactdal.Agregar(pCliente);

if (resultado > 0)

{

MessageBox.Show("Compra realizada Con Exito!!", "Guardado", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("No se pudo guardar el cliente", "Fallo!!", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

else

{

MessageBox.Show("Faltan campos por llenar", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

else

if (rdbUnidad.Checked)

{

if (cmbcolorunidad.Text != "" && cmbdiamunidad.Text != "")

{

//factura//

objeto = new Totales(Convert.ToInt32(nucantunidad.Value), Convert.ToInt32(cmbdiamunidad.Text), Convert.ToInt32(0), precioc, preciog, preciolip);

label29.Text = total.sumasubtotal().ToString();

ventasfact pCliente = new ventasfact();

if (rdfactura.Checked)

{

pCliente.tipo\_compra = "factura";

}

else

{

pCliente.tipo\_compra = "nota";

}

pCliente.fechaventa = dd.ToString("yyyy/MM/dd");

pCliente.Total = label29.Text;

int resultado = ventafactdal.Agregar(pCliente);

if (resultado > 0)

{

MessageBox.Show("Compra realizada Con Exito!!", "Guardado", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("No se pudo guardar el cliente", "Fallo!!", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

else

{

MessageBox.Show("Faltan campos por llenar", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

}

else

{

MessageBox.Show("Para realizar compras debe insertar los precios de los insumos", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

tabControl1.SelectTab(1);

}

}

else

{

MessageBox.Show("falta realizar compras");

}

}

private void button1\_Click(object sender, EventArgs e)

{

if (precioc != 0 && preciog != 0 && preciolip != 0)

{

if (rdbCaja.Checked)

{

if (cmbcolorcaja.Text != "" && cmbcajaunidades.Text != "" && cmbdiamcaja.Text != "")

{

objeto = new Totales(Convert.ToInt32(nucantcaja.Value), Convert.ToInt32(cmbdiamcaja.Text), Convert.ToInt32(cmbcajaunidades.Text), precioc, preciog, preciolip);

MessageBox.Show(objeto.total(2).ToString());

}

else

{ MessageBox.Show("Faltan campos por llenar", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation); }

}

else

if (rdbUnidad.Checked)

{

if (cmbcolorunidad.Text != "" && cmbdiamunidad.Text != "")

{

objeto = new Totales(Convert.ToInt32(nucantunidad.Value), Convert.ToInt32(cmbdiamunidad.Text), 0, precioc, preciog, preciolip);

MessageBox.Show(objeto.total(1).ToString());

}

else

{ MessageBox.Show("Faltan campos por llenar", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation); }

}

}

else

{

MessageBox.Show("Para realizar compras debe insertar los precios de los insumos", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

tabControl1.SelectTab(1);

}

}

private void txtlitrop\_KeyPress(object sender, KeyPressEventArgs e)

{

if (Char.IsDigit(e.KeyChar))

{

e.Handled = false;

}

else if (Char.IsControl(e.KeyChar))

{

e.Handled = false;

}

else if (Char.IsSeparator(e.KeyChar))

{

e.Handled = false;

}

else

{

MessageBox.Show("Solo se permiten Numeros", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

e.Handled = true;

}

}

private void txtnombre\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!(char.IsLetter(e.KeyChar)) && (e.KeyChar != (char)Keys.Back))

{

MessageBox.Show("Solo se permiten letras", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

e.Handled = true;

return;

}

}

private void cmbdiamcaja\_KeyPress(object sender, KeyPressEventArgs e)

{

if (Char.IsDigit(e.KeyChar))

{

e.Handled = false;

}

else if (Char.IsControl(e.KeyChar))

{

e.Handled = false;

}

else if (Char.IsSeparator(e.KeyChar))

{

e.Handled = false;

}

else

{

e.Handled = true;

}

if (!(char.IsLetter(e.KeyChar)) && (e.KeyChar != (char)Keys.Back))

{

e.Handled = true;

return;

}

}

private void rdbCaja\_CheckedChanged\_2(object sender, EventArgs e)

{

groupBox1.Enabled = false;

groupBox2.Enabled = true;

}

private void rdbUnidad\_CheckedChanged\_2(object sender, EventArgs e)

{

groupBox1.Enabled = true;

groupBox2.Enabled = false;

}

private void button1\_Click\_1(object sender, EventArgs e)

{

if (cmbventas.Text != "")

{

if (cmbventas.Text == "factura")

{

dataGridView3.DataSource = facturasdal.consultaventas();

}

if (cmbventas.Text == "nota")

{

dataGridView3.DataSource = facturasdal.consultaventasnot();

}

}

else

{

MessageBox.Show("porfavor seleccione un tipo de compra");

}

}

private void cmbventas\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void toolStripButton3\_Click(object sender, EventArgs e)

{

dvprintPreviewDialog1.Document = dvprintDocument1;

dvprintPreviewDialog1.ShowDialog();

}

private void dvprintDocument1\_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)

{

Bitmap bpm = Properties.Resources.nota;

Bitmap bmp= Properties.Resources.factura;

Image newImage=bmp;

if (label29.Text != "")

{

if (rdnota.Checked && rdbUnidad.Checked)

{

e.Graphics.DrawImage(bpm, 0, 0);

e.Graphics.DrawString(label18.Text, new Font("Arial", 10, FontStyle.Regular), Brushes.Black, new Point(94, 74));

e.Graphics.DrawString(label20.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(240, 75));

e.Graphics.DrawString("se vendieron esferas de color " + cmbcolorcaja.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(0, 130));

e.Graphics.DrawString("con diametro de " + cmbdiamcaja.Text +"cm la cantidad de esferas fue "+Convert.ToString(nucantunidad.Value), new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(0, 142));

e.Graphics.DrawString("y el total de la venta es " + label29.Text + " pesos", new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(0, 154));

}

if (rdbCaja.Checked && rdnota.Checked)

{

e.Graphics.DrawImage(bpm, 0, 0);

e.Graphics.DrawString(label18.Text, new Font("Arial", 10, FontStyle.Regular), Brushes.Black, new Point(94, 74));

e.Graphics.DrawString(label20.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(240, 75));

e.Graphics.DrawString("se vendio caja de esferas de color "+cmbcolorcaja.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(0,130));

e.Graphics.DrawString("con diametro de "+cmbdiamcaja.Text+"cm la caja contiene "+cmbcajaunidades.Text+"esferas", new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(0,142));

e.Graphics.DrawString("y el total de la venta es "+label29.Text +" pesos", new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(0, 154));

}

if (rdbCaja.Checked && rdfactura.Checked)

{

e.Graphics.DrawImage(bmp, 0, 0);

e.Graphics.DrawString("Caja Esferas", new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(5, 205));

e.Graphics.DrawString(cmbcajaunidades.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(205, 205));

e.Graphics.DrawString(cmbcolorcaja.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(81, 205));

e.Graphics.DrawString(cmbdiamcaja.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(150, 205));

e.Graphics.DrawString(label20.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(300, 205));

e.Graphics.DrawString(label18.Text, new Font("Arial", 10, FontStyle.Regular), Brushes.Black, new Point(80, 410));

}

else

if (rdbUnidad.Checked && rdfactura.Checked )

{

e.Graphics.DrawImage(bmp, 0, 0);

e.Graphics.DrawString("Esferas", new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(5, 205));

e.Graphics.DrawString(Convert.ToString(nucantunidad.Value), new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(205, 205));

e.Graphics.DrawString(cmbcolorunidad.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(81, 205));

e.Graphics.DrawString(cmbdiamunidad.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(150, 205));

e.Graphics.DrawString(label20.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(300, 205));

e.Graphics.DrawString(label18.Text, new Font("Arial", 10, FontStyle.Regular), Brushes.Black, new Point(80, 410));

}

}

else

{

MessageBox.Show("primero Termine la venta Gracias");

dvprintPreviewDialog1.Close();

}

}

private void button4\_Click\_1(object sender, EventArgs e)

{

precioc = carton;

preciog = gancho;

preciolip = pintura;

if (txtventas.Text != "")

{

txtventas.Enabled = false;

groupBox1.Enabled = true;

rdbUnidad.Enabled=true;

rdbCaja.Enabled = true;

button4.Enabled = false;

total = new subtotal(Convert.ToInt32(txtventas.Text));

MessageBox.Show("Listo");

}

}

private void button2\_Click\_1(object sender, EventArgs e)

{

// sub total//

dalsub sub = new dalsub();

int max=sub.id\_max();

DateTime dd = DateTime.Now;

if (rdbCaja.Checked)

{

if (cmbcolorcaja.Text != "" && cmbcajaunidades.Text != "" && cmbdiamcaja.Text != "")

{

objeto = new Totales(Convert.ToInt32(nucantcaja.Value), Convert.ToInt32(cmbdiamcaja.Text), Convert.ToInt32(cmbcajaunidades.Text), precioc, preciog, preciolip);

decimal total1 = Convert.ToDecimal(objeto.total(2));

label28.Text = total1.ToString();

total.agregar(Convert.ToDouble(total1));

sub.id\_factura = max + 1;

sub.Total = label28.Text;

sub.vendedor = label18.Text;

sub.producto = rdbCaja.Text;

sub.color = cmbcolorcaja.Text;

sub.unidades = cmbcajaunidades.Text;

sub.cantidad =Convert.ToString(nucantcaja.Value);

dalsub.Agregar(sub);

vent++;

}

else

{

MessageBox.Show("Faltan campos por llenar", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

else

if (rdbUnidad.Checked)

{

if (cmbcolorunidad.Text != "" && cmbdiamunidad.Text != "")

{

//factura//

objeto = new Totales(Convert.ToInt32(nucantunidad.Value), Convert.ToInt32(cmbdiamunidad.Text), Convert.ToInt32(0), precioc, preciog, preciolip);

label28.Text = objeto.total(1).ToString();

total.agregar(Convert.ToDouble(label28.Text));

sub.id\_factura = max + 1;

sub.Total = label28.Text;

sub.vendedor = label18.Text;

sub.producto = rdbUnidad.Text;

sub.color = cmbcolorunidad.Text;

sub.unidades = "0";

sub.cantidad = Convert.ToString(nucantunidad.Value);

dalsub.Agregar(sub);

vent++;

}

else

{

MessageBox.Show("Faltan campos por llenar", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

}

private void dataGridView3\_KeyPress(object sender, KeyPressEventArgs e)

{

MessageBox.Show("hola");

}

private void dataGridView3\_CellClick(object sender, DataGridViewCellEventArgs e)

{

Form2 frm = new Form2();

string valor=dataGridView3.CurrentRow.Cells[0].Value.ToString();

frm.datasubrorales.DataSource= subtot.consultasubtotales(valor);

frm.Show();

# Código Clase

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using MySql.Data.MySqlClient;

using System.Windows.Forms;

namespace Esferas

{

public class BdComun

{

//conexion a la base de datos

public static MySqlConnection ObtenerConexion()

{

MySqlConnection conectar = new MySqlConnection("server=db4free.net;database=esferas123;port=3306;Uid=cabresto5; pwd=cabresto5;");

conectar.Open();

return conectar;

}

}

//clase donde guardara los datos para asignarlos a la base de datos mediante un metodo

//se llama cliente pero es de los trbajadores

public class Cliente

{

public int Id { get; set; }

public string Nombre { get; set; }

public string Apellidop { get; set; }

public string apellidom { get; set; }

public string edad { get; set; }

public Cliente() { }

public Cliente(int pId, string pNombre, string pApellidopa, string apellidoma, string Edad)

{

this.Id = pId;

this.Nombre = pNombre;

this.Apellidop = pApellidopa;

this.apellidom = apellidoma;

this.edad = Edad;

}

}

//metodos de insertado y consultas

public class ClientesDAL

{

public static int Agregar(Cliente pCliente)

{

int retorno = 0;

MySqlCommand comando = new MySqlCommand(string.Format("Insert into Trabajador (nombre, apellido\_paterno, apellido\_materno, edad) values ('{0}','{1}','{2}', '{3}')",

pCliente.Nombre, pCliente.Apellidop, pCliente.apellidom, pCliente.edad), BdComun.ObtenerConexion());

retorno = comando.ExecuteNonQuery();

return retorno;

}

public static List<Cliente> Buscar()

{

List<Cliente> \_lista = new List<Cliente>();

MySqlCommand \_comando = new MySqlCommand(String.Format(

"SELECT \* FROM Trabajador"), BdComun.ObtenerConexion());

MySqlDataReader \_reader = \_comando.ExecuteReader();

while (\_reader.Read())

{

Cliente pCliente = new Cliente();

pCliente.Id = \_reader.GetInt32(0);

pCliente.Nombre = \_reader.GetString(1);

pCliente.Apellidop = \_reader.GetString(2);

pCliente.apellidom = \_reader.GetString(3);

pCliente.edad = \_reader.GetString(4);

\_lista.Add(pCliente);

}

return \_lista;

}

public static List<Cliente> nombres()

{

List<Cliente> \_lista = new List<Cliente>();

MySqlCommand \_comando = new MySqlCommand(String.Format(

"SELECT nombre FROM Trabajador"), BdComun.ObtenerConexion());

MySqlDataReader \_reader = \_comando.ExecuteReader();

while (\_reader.Read())

{

Cliente pCliente = new Cliente();

pCliente.Nombre = \_reader.GetString(0);

}

return \_lista;

}

}

public class ventasfact

{

public int id { get; set; }

public string tipo\_compra { get; set; }

public string fechaventa { get; set; }

public string Total { get; set; }

public ventasfact() { }

public ventasfact(int pid,string ven, string product, string tipo\_com, string colo, string unidad, string cant, string fecha,string tot)

{

this.id = pid;

this.tipo\_compra = tipo\_com;

this.fechaventa = fecha;

this.Total = tot;

}

}

public class ventafactdal

{

public static int Agregar(ventasfact ventas)

{

int retorno = 0;

MySqlCommand comando = new MySqlCommand(string.Format("Insert into factura (tipo\_compra,fecha\_venta, Total) values ('{0}','{1}','{2}')",

ventas.tipo\_compra, ventas.fechaventa, ventas.Total), BdComun.ObtenerConexion());

retorno = comando.ExecuteNonQuery();

return retorno;

}

}

public class Totales

{

public int Cantidad\_esfera { get; set; }

public int diametroesfera { get; set; }

public int unidades { get; set; }

public double precio\_pintura { get; set; }

public double precio\_carton { get; set; }

public double precio\_ganchos { get; set; }

public Totales() { }

public Totales(int Cantidad, int diamentro, int unidad, double precarton, double precioganchos, double preciopintu)

{

Cantidad\_esfera = Cantidad;

unidades = unidad;

diametroesfera = diamentro;

precio\_carton = precarton;

precio\_ganchos = precioganchos;

precio\_pintura = preciopintu;

}

public double total(int tipo\_venta)

{

double total = 0;

double ppintura;

double pgancho;

double pcarton;

double porcentaje;

if (tipo\_venta == 1)//tipo de venta por unidad

{

if (diametroesfera == 3)

{

ppintura = precio\_pintura \* 1 / 300;

pgancho = precio\_ganchos \* 1 / 700;

porcentaje = ppintura + pgancho;

porcentaje = porcentaje \* .30 + porcentaje;

total = porcentaje \* Cantidad\_esfera;

}

else

if (diametroesfera == 5)

{

ppintura = precio\_pintura \* 1 / 265;

pgancho = precio\_ganchos \* 1 / 700;

porcentaje = ppintura + pgancho;

porcentaje = porcentaje \* .30 + porcentaje;

total = porcentaje \* Cantidad\_esfera;

}

}

if (tipo\_venta == 2)//tipo de venta por caja

{

if (diametroesfera == 3 && unidades == 6)

{

ppintura = precio\_pintura \* 6 / 300;

pgancho = precio\_ganchos \* 6 / 700;

pcarton = precio\_carton \* 1 / 30;

porcentaje = ppintura + pgancho + pcarton;

porcentaje = porcentaje \* .30 + porcentaje;

total = porcentaje \* Cantidad\_esfera;

}

else

{

if (diametroesfera == 3 && unidades == 12)

{

ppintura = precio\_pintura \* 12 / 300;

pgancho = precio\_ganchos \* 12 / 700;

pcarton = precio\_carton \* 1 / 20;

porcentaje = ppintura + pgancho + pcarton;

porcentaje = porcentaje \* .30 + porcentaje;

total = porcentaje \* Cantidad\_esfera;

}

else

{

if (diametroesfera == 3 && unidades == 24)

{

ppintura = precio\_pintura \* 24 / 300;

pgancho = precio\_ganchos \* 24 / 700;

pcarton = precio\_carton \* 1 / 15;

porcentaje = ppintura + pgancho + pcarton;

porcentaje = porcentaje \* .30 + porcentaje;

total = porcentaje \* Cantidad\_esfera;

}

else

{

if (diametroesfera == 5 && unidades == 6)

{

ppintura = precio\_pintura \* 6 / 265;

pgancho = precio\_ganchos \* 6 / 700;

pcarton = precio\_carton \* 1 / 30;

porcentaje = ppintura + pgancho + pcarton;

porcentaje = porcentaje \* .30 + porcentaje;

total = porcentaje \* Cantidad\_esfera;

}

else

{

if (diametroesfera == 5 && unidades == 12)

{

ppintura = precio\_pintura \* 12 / 265;

pgancho = precio\_ganchos \* 12 / 700;

pcarton = precio\_carton \* 1 / 20;

porcentaje = ppintura + pgancho + pcarton;

porcentaje = porcentaje \* .30 + porcentaje;

total = porcentaje \* Cantidad\_esfera;

}

else

{

if (diametroesfera == 5 && unidades == 24)

{

ppintura = precio\_pintura \* 24 / 265;

pgancho = precio\_ganchos \* 24 / 700;

pcarton = precio\_carton \* 1 / 15;

porcentaje = ppintura + pgancho + pcarton;

porcentaje = porcentaje \* .30 + porcentaje;

total = porcentaje \* Cantidad\_esfera;

}

}

}

}

}

}

}

return Math.Round(total);

}

}

public class insumos

{

public int Id { get; set; }

public string precio\_carton { get; set; }

public string precio\_ganchos { get; set; }

public string precio\_pintura { get; set; }

public insumos() { }

public insumos(int pId, string carton, string gancho, string pintura)

{

this.Id = pId;

this.precio\_carton = carton;

this.precio\_ganchos = gancho;

this.precio\_pintura = pintura;

}

}

public class insumosdad

{

public static int Agregar(insumos ventas)

{

int retorno = 0;

MySqlCommand comando = new MySqlCommand(string.Format("Insert into Insumos (precio\_carton,precio\_gancho,precio\_pintura) values ('{0}','{1}','{2}')",

ventas.precio\_carton, ventas.precio\_ganchos, ventas.precio\_pintura), BdComun.ObtenerConexion());

retorno = comando.ExecuteNonQuery();

return retorno;

}

public static List<insumos> cosultainsu()

{

List<insumos> \_lista = new List<insumos>();

MySqlCommand \_comando = new MySqlCommand(String.Format(

"SELECT \* FROM Insumos"), BdComun.ObtenerConexion());

MySqlDataReader \_reader = \_comando.ExecuteReader();

while (\_reader.Read())

{

insumos pCliente = new insumos();

pCliente.Id = \_reader.GetInt32(0);

pCliente.precio\_carton = \_reader.GetString(1);

pCliente.precio\_ganchos = \_reader.GetString(2);

pCliente.precio\_pintura = \_reader.GetString(3);

\_lista.Add(pCliente);

}

return \_lista;

}

}

public class totalesdalfact

{

public static int Agregar(ventasfact ventas)

{

int retorno = 0;

MySqlCommand comando = new MySqlCommand(string.Format("Insert into Insumos (vendedor,producto,tipo\_compra,color,unidades,cantidad,fecha\_venta,total) values ('{0}','{1}','{2}','{3}','{4}','{5}','{6}','{7}')",

ventas.tipo\_compra,ventas.fechaventa,ventas.Total), BdComun.ObtenerConexion());

retorno = comando.ExecuteNonQuery();

return retorno;

}

}

public class facturasdal

{

public static List<ventasfact> consultaventas()

{

List<ventasfact> \_lista = new List<ventasfact>();

MySqlCommand \_comando = new MySqlCommand(String.Format(

"SELECT \* FROM factura where tipo\_compra='factura'"), BdComun.ObtenerConexion());

MySqlDataReader \_reader = \_comando.ExecuteReader();

while (\_reader.Read())

{

ventasfact pCliente = new ventasfact();

pCliente.id = \_reader.GetInt32(0);

pCliente.tipo\_compra = \_reader.GetString(1);

pCliente.fechaventa = \_reader.GetString(2);

pCliente.Total = \_reader.GetString(3);

\_lista.Add(pCliente);

}

return \_lista;

}

public static List<ventasfact> consultaventasnot()

{

List<ventasfact> \_lista = new List<ventasfact>();

MySqlCommand \_comando = new MySqlCommand(String.Format(

"SELECT \* FROM factura where tipo\_compra='nota'"), BdComun.ObtenerConexion());

MySqlDataReader \_reader = \_comando.ExecuteReader();

while (\_reader.Read())

{

ventasfact pCliente = new ventasfact();

pCliente.id = \_reader.GetInt32(0);

pCliente.tipo\_compra = \_reader.GetString(1);

pCliente.fechaventa = \_reader.GetString(2);

pCliente.Total = \_reader.GetString(3);

\_lista.Add(pCliente);

}

return \_lista;

}

}

public class subtotal

{

private int tam;

private double[] sub\_total ;

int pos;

public subtotal() { }

public subtotal(int t)

{

sub\_total = new double [t];

tam = t;

}

public void agregar(double dato)

{

if (pos < tam)

{

sub\_total[pos] = dato;

pos++;

}

else

{

MessageBox.Show("llego al limite de Ventas que usted selecciono Termine La compra abajo");

}

}

public double sumasubtotal()

{

double total2=0;

for(int i=0;i<tam;i++)

{

total2 += sub\_total[i];

}

return total2;

}

////////////////////////////////clase subtotales///////

}

public class dalsub

{

public int subtotal1 { get; set; }

public int id\_factura { get; set; }

public string vendedor { get; set; }

public string producto { get; set; }

public string color { get; set; }

public string unidades { get; set; }

public string cantidad { get; set; }

public string fechaventa { get; set; }

public string Total { get; set; }

public dalsub() { }

public dalsub(int subtot,int id\_factura,int pid, string ven, string product, string colo, string unidad, string cant,string tot)

{

this.subtotal1 =subtot;

this.id\_factura = id\_factura;

this.vendedor = ven;

this.producto = product;

this.color = colo;

this.unidades = unidad;

this.cantidad = cant;

this.Total = tot;

}

public static int Agregar(dalsub subtot)

{

int retorno = 0;

MySqlCommand comando = new MySqlCommand(string.Format("Insert into subtotal (id\_factura,vendedor,producto,color,unidades,cantidad,subtotal) values ('{0}','{1}','{2}','{3}','{4}','{5}','{6}')",

subtot.id\_factura, subtot.vendedor, subtot.producto, subtot.color, subtot.unidades,subtot.cantidad, subtot.Total), BdComun.ObtenerConexion());

retorno = comando.ExecuteNonQuery();

return retorno;

}

public int id\_max()

{

int max;

MySqlCommand \_comando = new MySqlCommand(String.Format(

"SELECT max(id\_factura) FROM factura"), BdComun.ObtenerConexion());

max =(Int32)\_comando.ExecuteScalar();

return max;

}

}

public class subtot

{

public string vendedor { get; set; }

public string producto { get; set; }

public string color { get; set; }

public string unidades { get; set; }

public string cantidad { get; set; }

public string Total { get; set; }

public subtot() { }

public subtot(int subtot,int id\_factura,int pid, string ven, string product, string colo, string unidad, string cant,string tot)

{

this.vendedor = ven;

this.producto = product;

this.color = colo;

this.unidades = unidad;

this.cantidad = cant;

this.Total = tot;

}

public static List<subtot> consultasubtotales(String dato)

{

List<subtot> \_lista = new List<subtot>();

MySqlCommand \_comando = new MySqlCommand(String.Format(

"SELECT \* FROM subtotal where id\_factura=" + dato), BdComun.ObtenerConexion());

MySqlDataReader \_reader = \_comando.ExecuteReader();

while (\_reader.Read())

{

subtot pCliente = new subtot();

pCliente.vendedor = \_reader.GetString(2);

pCliente.producto = \_reader.GetString(3);

pCliente.color = \_reader.GetString(4);

pCliente.unidades = \_reader.GetString(5);

pCliente.cantidad = \_reader.GetString(6);

pCliente.Total = \_reader.GetString(7);

\_lista.Add(pCliente);

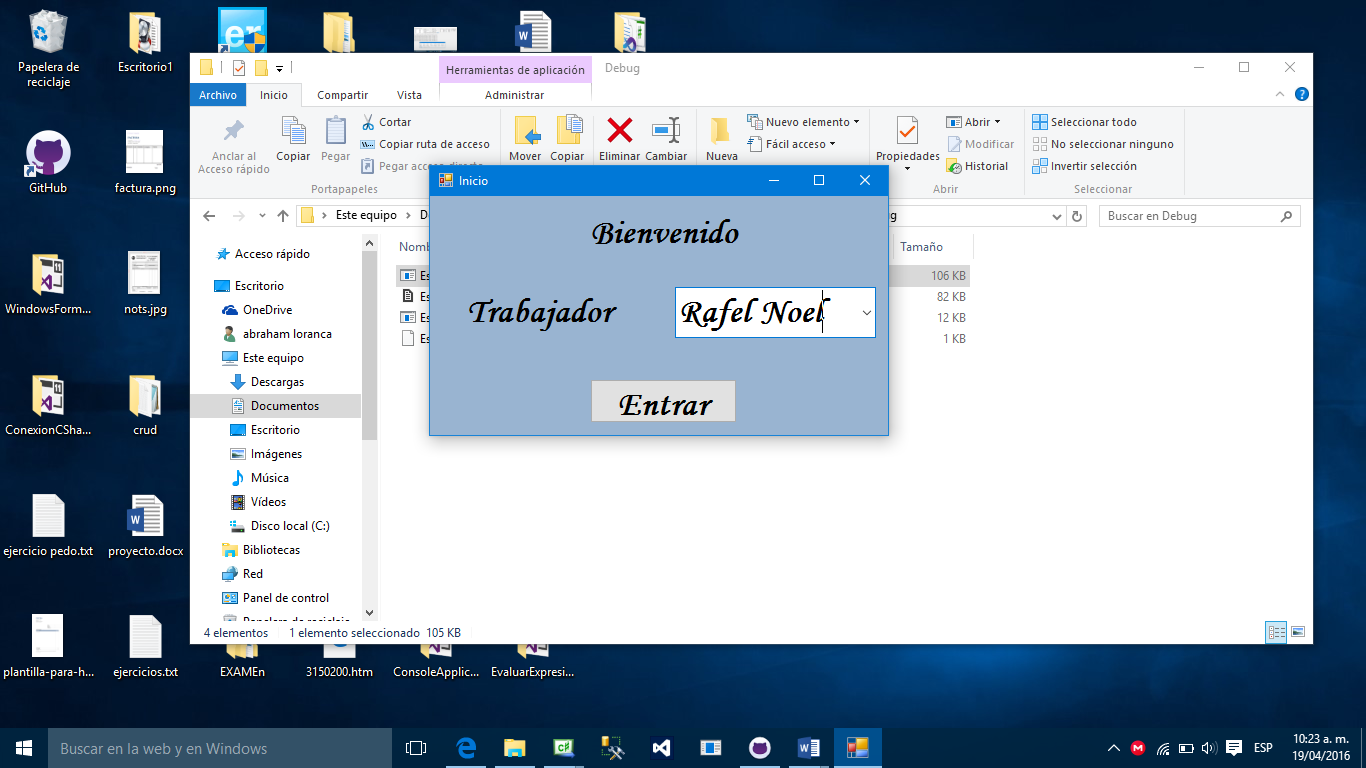
}

return \_lista;

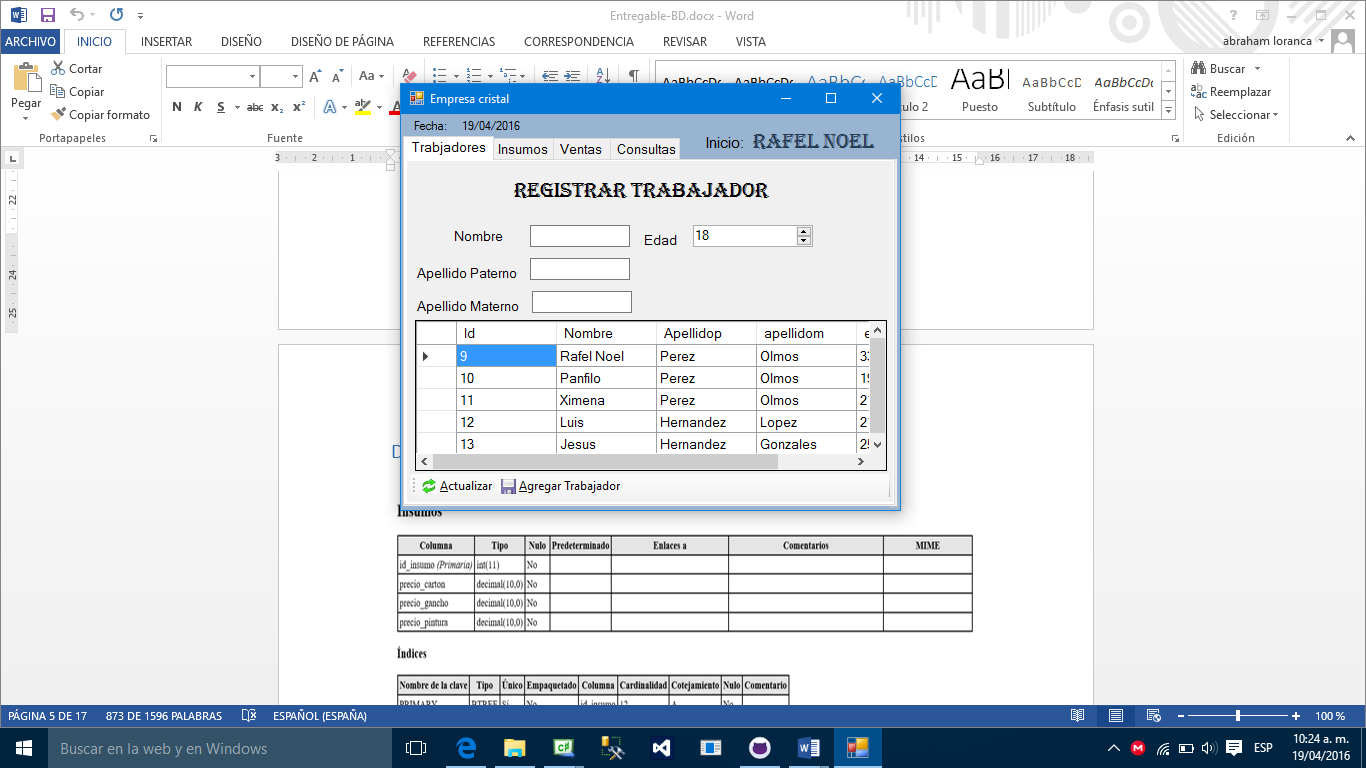
}

# Capturas del programa corriendo desde el .exe

Pantalla de inicio



En la siguiente pantalla podemos ver que muestra la fecha y el trabajador que esta atendiendo se vacían los datos de la base de datos en los datagriedview con una consulta simple



En la pantalla consultas se vacían las comprar realizadas si se da clic en alguna compra se mostrara a detalle la compra por ejemplo ; lo que el cliente compro y si el cliente compro varios producto

