

**UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE**



**INSTRUCTORA**

**INTEGRANTES:**

**Grupo:**

**Nrc:**

**Carrera:**

Ing. Tatiana Gualotuña

Kleber Enrique Chavez Oscullo

Jeimy Marley Morales Sosa

Camilo Andrés Orrico Emme

N° 1

4681

Software

Contenido

[Título del Proyecto: Escoge tu sabor 3](#_Toc109175845)

[Funcionalidades 4](#_Toc109175846)

[Reglas: 5](#_Toc109175847)

[2) Casos de uso 7](#_Toc109175848)

[3) Descripción de casos de uso 7](#_Toc109175849)

[4) Diagrama de clases 1](#_Toc109175850)4

[5) Código 1](#_Toc109175851)6

[6) Captura de la interfaz grafica ..4530](#_Toc109175852)

[7 captura de la pantalla de la creación de datos 471](#_Toc109175853)

[8 captura de pantallas corridas 481](#_Toc109175854)

**Medidas de ingredientes para un restaurante**

# Título del Proyecto: Escoge tu sabor

El programa que se planea realizar, el usuario va ser el empleado del restaurante, el software va consistir de diferentes tipos de interfases con el usuario, una pantalla de inicio que tiene la opción de iniciar sesión y registrase, en la pantalla de iniciar sesión el usuario dispondrá de deberá introducir sus credenciales (Id “cedula” y contraseña) en el registrar el usuario puede crear su cuenta donde deberá introducir datos personales (Nombre, apellido, fecha de nacimiento, Id, e-mail, numero celular y clave).

Dentro del programa

El menú debe aparecer después de iniciar sesión y en esta pantalla se debe mostrar las siguientes funciones:

Ver perfil: información del usuario, el usuario puede editar y guardar sus datos a excepción de su id (cedula).

Historial: se muestra el historial de pedidos vendidos en ese día.

Preguntas: se puede realizar preguntas o comentarios sobre la utilidad de nuestra aplicación

Cuadro de texto con el número del pedido: este servirá al ingresar con el fin de mantener el mismo número de pedido l ingresar la orden

Bebidas: nos redirigirá a otra pantalla donde se debe presentar el mismo numero de orden y con las opciones disponibles que tiene de bebidas, la opción de agregar a la orden y regresar al menú principal

Platos de entrada: nos redirigirá a otra pantalla donde se debe presentar el mismo número de orden y con las opciones disponibles que tiene de entradas, la opción de agregar a la orden y regresar al menú principal

Platos fuerte: nos redirigirá a otra pantalla donde se debe presentar el mismo número de orden y con las opciones disponibles que tiene de platos fuertes, la opción de agregar a la orden y regresar al menú principal

Postre: nos redirigirá a otra pantalla donde se debe presentar el mismo número de orden y con las opciones disponibles que tiene de postres, la opción de agregar a la orden y regresar al menú principal

Cuando ya se registró todo la orden se añade todo a una pantalla de confirmación del pedido donde se agregar el nombre del cliente para la facturación, detalla el costo, y se puede eliminar los productos que el cliente no desee antes de confirmar la orden.

Al momento de aceptar la orden, se muestra la factura con todos los datos de la orden y el precio total de la orden, en esta pantalla tendrá disponible la opción de regresar al menú principal para realizar otra orden, imprimir, y visualizar los ingredientes que se deben utilizar para ese pedido.

**Requerimientos funcionales**

* Ventana principal del aplicativo.
* Ingresar al sistema con un id (cedula) y una clave, si el usuario aún no se encuentra registrado en el sistema, puede optar por la opción crear cuenta de usuario.
* Inicio del sistema, dentro del menú.
  + Opciones de las categorías en la que están divididos cada platillo
    - Bebidas
    - Platos de entrada
    - Platos fuertes
    - Postres
  + Ver perfil
  + Historial de pedidos
  + Ayuda
  + Salir
* Dentro del apartado categoría se va a mostrar los platillos disponibles dependiendo de las categorías, cada platillo estará conformado con, nombre, precio e imagen, al escoger el platillo se ira guardando la información, para el usuario (precio, cantidad y nombre del platillo) para el chef (nombre del platillo y los ingredientes que se utilizaran).
* Al ya haber escogido cada uno de los platillos, tendrá la opción confirmar pedido , en esta ventana se agrega el nombre del cliente y se va a mostrar los platillos escogidos, cantidad, el precio y el total a apagar por cada producto escogido.
* Aceptar los pedidos, se procesará la factura con el nombre del cliente y el precio total a pagar, estarán disponibles el botón regresar al menú ver ingredientes e imprimir factura,
* Regresa al menú para comenzar con el proceso nuevamente

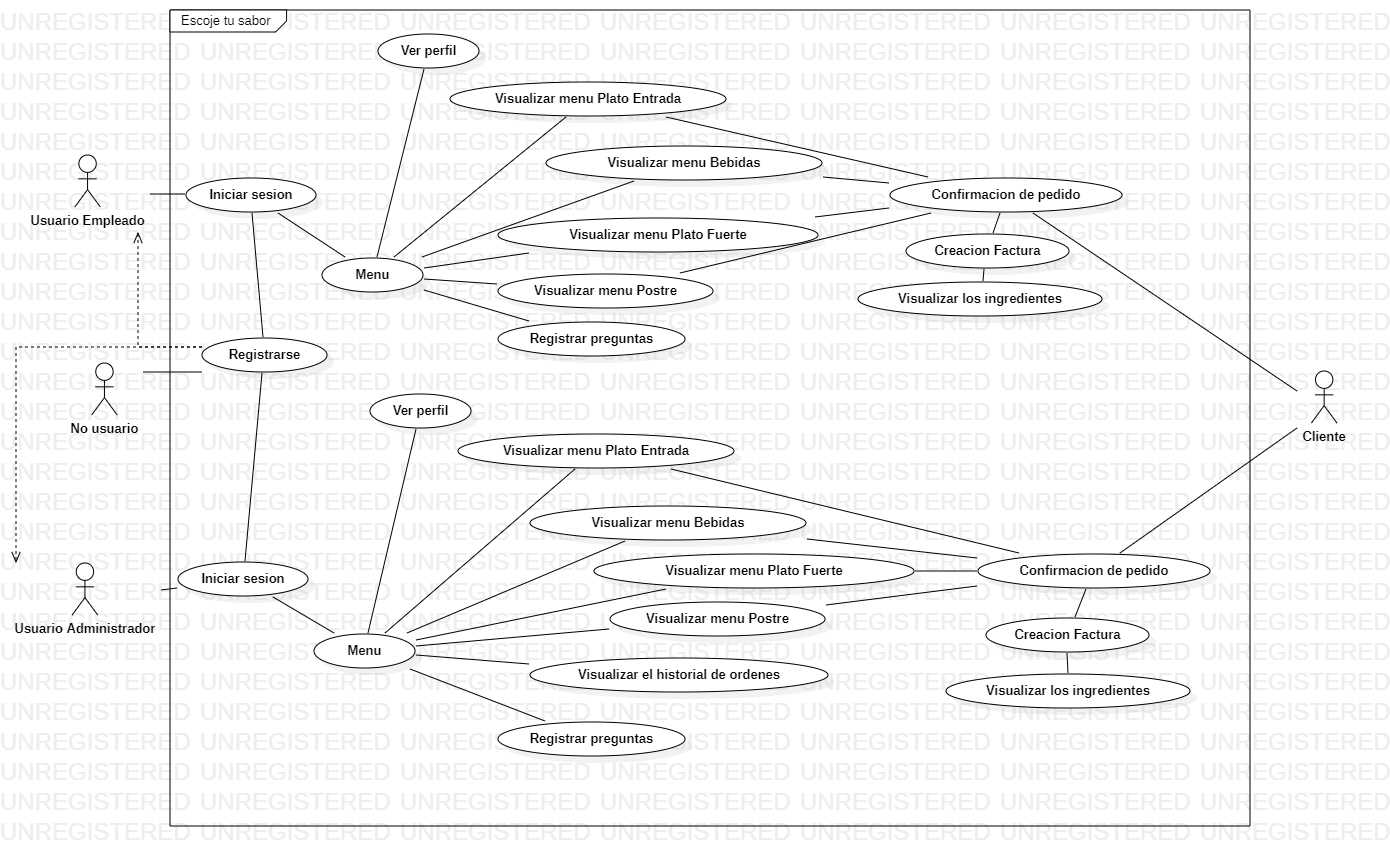
# Funcionalidades

* + - 1. Pantalla de inicio
      2. Crear una cuenta o iniciar sesión.
      3. Registrar la cuenta nueva e iniciar sesión.
      4. Iniciar sesión o regresar a pantalla principal.
      5. Menú principal.
         * Ver información de perfil
         * Ingresar a preguntas
         * Ver versión
         * Ver historial
         * Ver bebidas
         * Ver platos de entrada
         * Ver platos fuertes
         * Ver postres
         * Ir a confirmación de pedido
      6. Información de perfil
         * Editar
         * Guardar
         * Menú Principal
      7. Preguntas
         * Registrar pregunta o comentario.
      8. Versión
      9. Historial de todas las ordenes
      10. Bebidas opciones
          * Agregar bebidas a la orden
          * Regresar a menú principal
      11. Platos de entrada opciones
          * Agregar entradas a la orden
          * Regresar a menú principal
      12. Platos fuertes opciones
          * Agregar platos fuertes a la orden
          * Regresar a menú principal
      13. Postres opciones
          * Agregar postres a la orden
          * Regresar a menú principal
      14. Confirmación del pedido
          * Eliminar elemento de la orden
          * Ingresar nombre del cliente para la factura
          * Aceptar orden
      15. Factura
          * Imprimir factura
          * Mostrar ingredientes
          * Mostrar total
          * Regresar a menú principal

# Reglas:

1. **Registrar:** Los datos personales serán guardados para la validación en el registro para lo cual deberá introducir, su primer nombre que constara solo de letras minúsculas o mayúsculas sin caracteres especiales ni números, su primer apellido que constara solo de letras minúsculas o mayúsculas sin caracteres especiales ni números, id (cedula), contraseña, verificación de la contraseña, email aquí no existen restricciones, número de celular y fecha de nacimiento.
2. **Login:** el id va a constar de 10 dígitos los cuales corresponderán a el número de cedula sin caracteres especiales ni letras. En la contraseña no existirá restricción alguna para la seguridad del usuario.
3. **Visualizar menú principal**
   * Seleccionar categoría (Bebidas), dirige al usuario de servicio a un submenú de la clase bebida
   * Seleccionar categoría (Platos de entrada) dirige al usuario de servicio a un submenú de la clase plato de entrada
   * Seleccionar categoría (Platos Fuertes) dirige al usuario de servicio a un submenú del plato fuerte
   * Seleccionar categoría (postres) dirige al usuario de servicio a un submenú del postre
   * **Ver perfil:** Los usuarios podrán ver sus datos personales ingresados en la creación de su cuenta los cuales podrá modificar a excepción del id.
   * **Ver historial:** Se podrá observar todas las facturas realizadas con los detalles de la orden.
   * **Ver preguntas:** Se podrá registrar comentarios y preguntas
4. **Visualizar el submenú de la categoría** 
   * Nombre del platillo, para cada una de las categorías (bebidas, plato de entrada, plato fuerte y postres)
   * Precio de cada platillo, para cada una de las categorías (bebidas, plato de entrada, plato fuerte y postres)
   * Imagen de referencia sobre el platillo, para cada una de las categorías (bebidas, plato de entrada, plato fuerte y postres)
5. **Para los usuarios (servicio),** 
   * **Ver pantalla con los platillos escogidos,** 
     + Nombre y apellido del cliente
     + Numero de orden
     + Cantidad
     + Precio
     + Total, a pagar
   * **Eliminar pedido:** el comensal tiene la opción que el personal de servicio elimine el pedido.
   * **Cancelar pedido:** el comensal tiene la opción que el personal de servicio cancele algún producto del pedido
   * **Aceptar los pedidos:** al guardad el pedido se le mostrara al personal de cocina para empezar la elaboración de la orden
6. **Visualizar la información del pedido**
   * **Información:** Nombre del cliente, numero de orden y el lugar de la mesa.
     + **Llevará a la facturación** al mostrar la facturación en el total de la orden, la información del cliente Nombre
7. **Visualizar los ingredientes.** 
   * **Usuario** 
     + Visualizar las recetas y la cantidad de cada ingrediente para la realización del platillo
8. **Visualizar el historial de órdenes.**
   * **Usuario (personal administrativo)**
     + Visualiza las ordenes que se hicieron en ese día para llevar un registro de control.

# 2) Casos de uso



# 3) Descripción de casos de uso

|  |  |
| --- | --- |
| **Nombre:** | Registrar |
| **Autor:** | Camilo Orrico |
| **Fecha:** | 3/06/2022 |
| **Descripción:**  El usuario puede crear una cuenta en el sistema ingresando sus datos personales (id “cedula”, contraseña, nombre, apellido, e-mail, número celular). | |
| **Actores:**  Usuario de servicio  Usuario administrador | |
| **Precondiciones:**  Ser un empleado del restaurante | |
| **Flujo Normal:**  El usuario debe introducir sus datos personales (primer nombre, primer apellido, email,  Numero de celular, id “cedula”).  El usuario creara una contraseña, para crear la cuenta junto con los datos anteriores. | |
| **Flujo alternativo:**  El usuario puede cancelar el registro  El usuario puede entrar al login desde el registrar | |
| **Postcondiciones:**  El usuario podrá ingresar en el sistema con su id y contraseña | |

|  |  |
| --- | --- |
| **Nombre:** | Login |
| **Autor:** | Camilo Orrico |
| **Fecha:** | 3/06/2022 |
| **Descripción:**  Puede ingresar al sistema si se tiene una cuenta creada con anterioridad usando  Un id y contraseña registrada. | |
| **Actores:**  Usuario de servicio  Usuario administrador | |
| **Precondiciones:**  El usuario debe haber creado una cuenta con anterioridad. | |
| **Flujo Normal:**  El usuario debe introducir las credenciales (id y contraseña) tal cual como haya registrado los datos en el sistema | |
| **Flujo alternativo:**  El usuario puede cambiar de contraseña | |
| **Postcondiciones:**  El usuario ingresara en el menú principal, puede moverse entre submenús disponibles dentro del sistema y  usar las funcionalidades del sistema libremente | |

|  |  |
| --- | --- |
| **Nombre:** | Ver perfil |
| **Autor:** | Camilo Orrico |
| **Fecha:** | 3/06/2022 |
| **Descripción:**  Puede ingresar a un submenú, observar sus datos registrados y modificarlos a excepción de su id. | |
| **Actores:**  Usuario de servicio  Usuario administrador | |
| **Precondiciones:**  Haber creado una cuenta, haber iniciado sesión y entrado en el sistema desde el menú principal. | |
| **Flujo Normal:**  El usuario debe presionar en un submenú llamado información el cual se va a encontrar en la parte superior del menú.  El usuario podrá observar los datos registrados de su cuenta  El usuario podrá modificar los datos de su cuenta a excepción de su id. | |
| **Flujo alternativo:**  Regresar al menú principal | |
| **Postcondiciones:**  El usuario podrá continuar con la visualización del menú y su derivaciones | |

|  |  |
| --- | --- |
| **Nombre:** | Menú Principal |
| **Autor:** | Morales Marley |
| **Fecha:** | 18/07/2022 |
| **Descripción:**  Dentro del menú tenemos las opciones de ver perfil, ayuda, sobre el sistema, salir, historial, bebidas, platos fuertes, platos entrada, postres y confirmar pedido. | |
| **Actores:**  Usuario de servicio (toma la orden al cliente)  Usuario administrador  Clientes | |
| **Precondiciones:**  Ser un usuario de servicio  Estar registrado en el sistema  Haber iniciado sesión | |
| **Flujo Normal:**  El usuario de servicio, se encargará de tomar las ordenes de los clientes  El usuario de servicio, entrara a la categoría que el cliente desee, tiene cuatro tipos de categorías:   * Bebidas * Plato de entrada * Plato fuerte * Postres | |
| **Flujo alternativo:**  En la parte superior de la pantalla dispondrá de pestañas donde estará disponible (sobre el sistema, ayuda, preguntas frecuentes e historial de las ordenes) | |
| **Postcondiciones:**  El usuario de servicio puede comenzar a escoger los platillos que el cliente desee | |

|  |  |
| --- | --- |
| **Nombre:** | Visualizar submenú de la categoría |
| **Autor:** | Morales Marley |
| **Fecha:** | 03/07/2022 |
| **Descripción:**  En el submenú dependiendo en la categoría en la que se encuentre, se despliega cada uno de los platillos u opciones que estén disponibles con imágenes, nombre del platillo y precio. | |
| **Actores:**  Usuario de servicio  Cliente | |
| **Precondiciones:**  Seleccionar la categoría a la que desea entrar | |
| **Flujo Normal:**  El usuario de servicio, acoge las opciones que el cliente desee  El usuario de servicio tiene la opción de aceptar para que se guarden las opciones que el cliente escogió | |
| **Flujo alternativo:**  El usuario de servicio puede regresar al menú principal si el cliente decide no ordenar. | |
| **Postcondiciones:**  Visualizara cada una de las opciones que el cliente pidió en su orden | |

**Explicación de cada submenú de categorías**

|  |  |
| --- | --- |
| **Nombre:** | Visualizar submenú de bebidas |
| **Autor:** | Morales Marley |
| **Fecha:** | 19/07/2022 |
| **Descripción:**  Al escoger la opción de bebidas se presentan todas las opciones disponibles que tiene el restaurante para e cliente | |
| **Actores:**  Usuario de servicio  Usuario administrador  Clientes | |
| **Precondiciones:**  Escoger la opción bebidas del menú | |
| **Flujo Normal:**  El usuario de servicio, se encargará de tomar las ordenes de los clientes  El usuario de servicio, entrara a la categoría que el cliente desee, puede regresar nuevamente al menú para escoger otra categoría | |
| **Flujo alternativo:**  El cliente termino de ordenar, el usuario de servicio tiene la opción de confirmar pedidos donde se muestran todos los pedidos hechos por el cliente | |
| **Postcondiciones:**  El usuario de servicio tiene la opción de confirmar pedido o escoger un producto de otra categoría | |

|  |  |
| --- | --- |
| **Nombre:** | Visualizar submenú de platos de entrada |
| **Autor:** | Morales Marley |
| **Fecha:** | 19/07/2022 |
| **Descripción:**  Al escoger la opción de platos de entrada se presentan todas las opciones disponibles que tiene el restaurante para el cliente | |
| **Actores:**  Usuario de servicio  Usuario administrador  Clientes | |
| **Precondiciones:**  Escoger la opción platos de entrada del menú principal | |
| **Flujo Normal:**  El usuario de servicio, se encargará de tomar las ordenes de los clientes  El usuario de servicio, entrara a la categoría que el cliente desee, puede regresar nuevamente al menú para escoger otra categoría | |
| **Flujo alternativo:**  El cliente termino de ordenar, el usuario de servicio tiene la opción de confirmar pedidos donde se muestran todos los pedidos hechos por el cliente | |
| **Postcondiciones:**  El usuario de servicio tiene la opción de confirmar pedido o escoger un producto de otra categoría | |

|  |  |
| --- | --- |
| **Nombre:** | Visualizar submenú de platos fuertes |
| **Autor:** | Morales Marley |
| **Fecha:** | 19/07/2022 |
| **Descripción:**  Al escoger la opción de platos fuertes se presentan todas las opciones disponibles que tiene el restaurante para e cliente | |
| **Actores:**  Usuario de servicio  Usuario administrador  Clientes | |
| **Precondiciones:**  Escoger la opción platos fuertes del menú | |
| **Flujo Normal:**  El usuario de servicio, se encargará de tomar las ordenes de los clientes  El usuario de servicio, entrara a la categoría que el cliente desee, puede regresar nuevamente al menú para escoger otra categoría | |
| **Flujo alternativo:**  El cliente termino de ordenar, el usuario de servicio tiene la opción de confirmar pedidos donde se muestran todos los pedidos hechos por el cliente | |
| **Postcondiciones:**  El usuario de servicio tiene la opción de confirmar pedido o escoger un producto de otra categoría | |

|  |  |
| --- | --- |
| **Nombre:** | Visualizar submenú de postres |
| **Autor:** | Morales Marley |
| **Fecha:** | 19/07/2022 |
| **Descripción:**  Al escoger la opción de postres se presentan todas las opciones disponibles que tiene el restaurante para e cliente | |
| **Actores:**  Usuario de servicio  Usuario administrador  Clientes | |
| **Precondiciones:**  Escoger la opción postres del menú | |
| **Flujo Normal:**  El usuario de servicio, se encargará de tomar las ordenes de los clientes  El usuario de servicio, entrara a la categoría que el cliente desee, puede regresar nuevamente al menú para escoger otra categoría | |
| **Flujo alternativo:**  El cliente termino de ordenar, el usuario de servicio tiene la opción de confirmar pedidos donde se muestran todos los pedidos hechos por el cliente | |
| **Postcondiciones:**  El usuario de servicio tiene la opción de confirmar pedido o escoger un producto de otra categoría | |

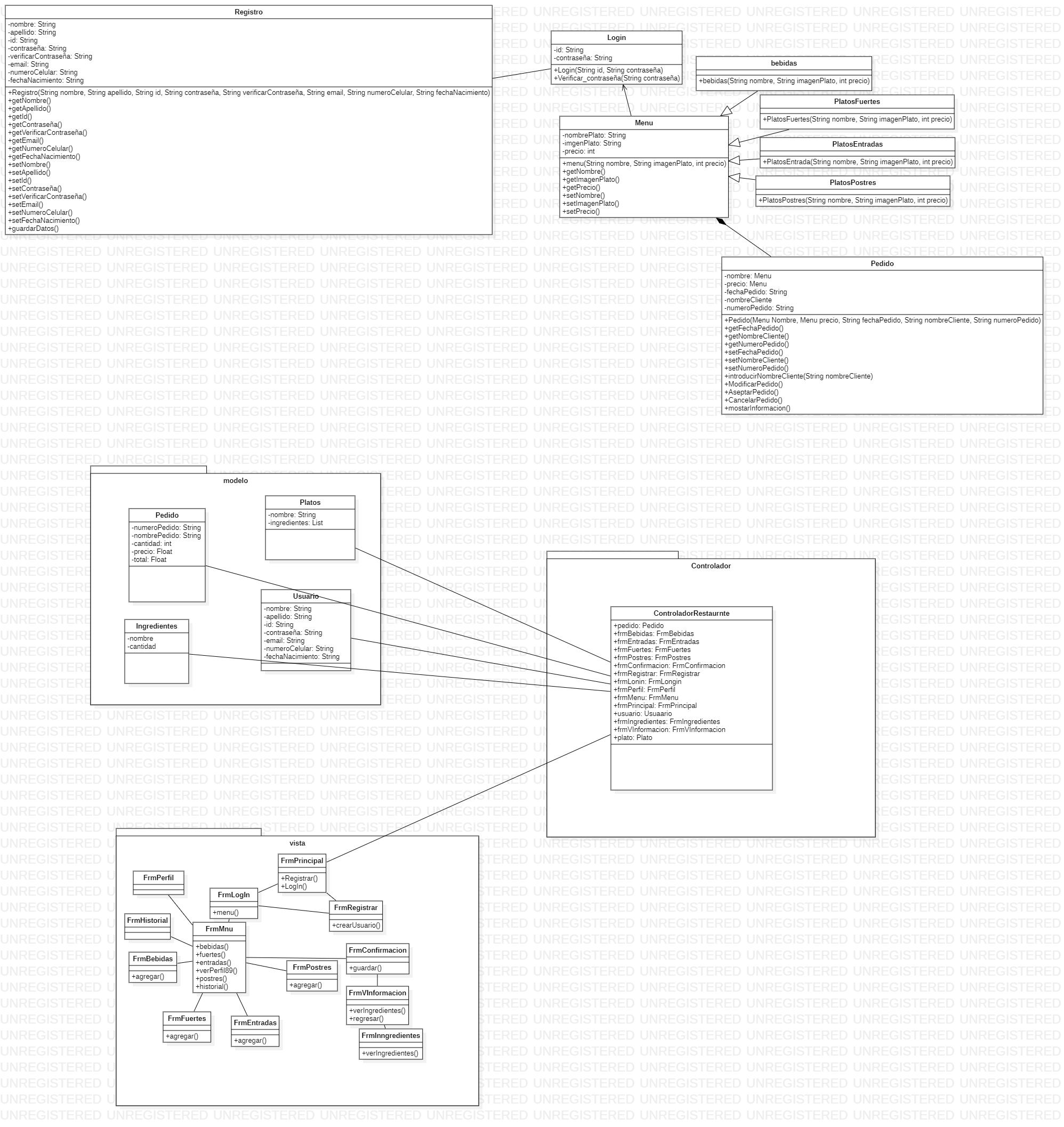
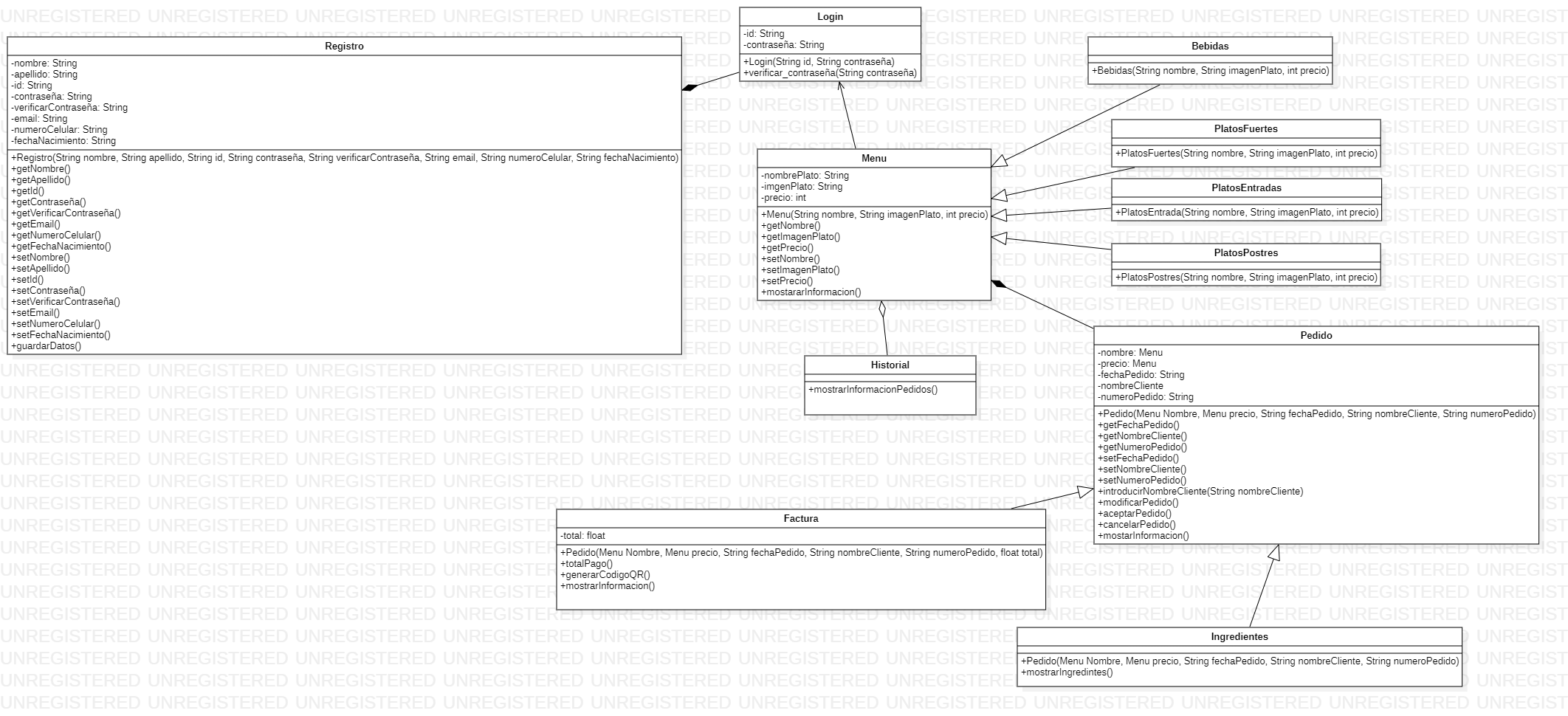
|  |  |
| --- | --- |
| **Nombre:** | Confirmación de platillos |
| **Autor:** | Morales Marley |
| **Fecha:** | 19/07/2022 |
| **Descripción:**  El usuario de servicio, al acabar de tomar la orden del comensal, realizara la confirmación del pedido. | |
| **Actores:**  Usuario de servicio y cliente | |
| **Precondiciones:**  Haber seleccionado los platillos que el cliente deseaba. | |
| **Flujo Normal:**  El usuario de servicio confirma cada uno de los platillos que el comensal ordeno  Al no ser así, tendrá una opción de eliminar un producto o cancelar la orden. | |
| **Flujo alternativo:**  El cliente cancela la orden, al cancelar la orden regresara al menú principal | |
| **Postcondiciones:**  Al ser confirmada la orden, se Procesa la factura y lista de ingredientes. | |

|  |  |
| --- | --- |
| **Nombre:** | Visualizar la información del pedido |
| **Autor:** | Kleber Chavez |
| **Fecha** | 02/06/2022 |
| **Descripción**:  Permite la visualización de los datos de la factura, como el nombre del comensal los datos del mismo, el precio por platillo, total de la cuenta y un código QR | |
| **Actores**: Usuario de servicio | |
| **Precondiciones**: Registró de pago del valor del pedido | |
| **Flujo normal:**  1) El Usuario de servicio, confirma el pago echo por el comensal.  3) El Usuario de servicio, guarda la factura y posterior impresión del mismo que mostrara el código QR  4) El Usuario administrativo o de servicio, entrega la factura al comensal. | |
| **Postcondiciones:**  Se visualiza los detalles de la preparación del platillo | |

|  |  |
| --- | --- |
| **Nombre:** | Visualizar los ingredientes |
| **Autor:** | Kleber Chavez |
| **Fecha:** | 02/06/2022 |
| **Descripción:**  Ve el detalle de todos los ingredientes por plato y las cantidades a usar para la elaboración del mismo | |
| **Actores**: Usuario de cocina | |
| Precondiciones:   La orden echa por el cliente debe ser confirmada. | |
| **Flujo normal:**  1. El sistema muestra el pedido especificando por categoría.  2. El sistema muestra cantidad de ingredientes por plato a utilizar  3. El sistema muestra la suma a utilizar por integrantes. | |
| **Postcondiciones:**  El usuario de cocina, envía el pedido finalizado | |

|  |  |
| --- | --- |
| **Nombre:** | Visualizar el historial de órdenes |
| **Autor:** | Kleber Chavez |
| **Fecha:** | 02/06/2022 |
| **Descripción**:  Permite al usuario administrativo visualiza el historial de las ordenes echas ese día, | |
| **Actores**:  Usuario administrativo | |
| **Precondiciones**:   Estar registrado en el sistema como Administrativo | |
| **Flujo normal:**  En una pestaña del menú estará disponible la opción Historial, si es un usuario administrativo puede entrar a realizar un seguimiento de las ordenes echas | |
| **Flujo Alternativo:**  Al no ser un Usuario Administrador, no podrá ver el historial. | |
| **Postcondiciones:**  Regreso al menú principal | |

# 4) Diagrama de clases



# 5) Código

**Controlador**

Clase controlador restaurante

package controlador;

import vista.FrmConfirmacion;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.KeyEvent;

import java.awt.event.KeyListener;

import static java.lang.String.valueOf;

import java.util.ArrayList;

import javax.swing.JTable;

import javax.swing.table.DefaultTableModel;

import modelo.Pedido;

import modelo.PedidosDAO;

import vista.FrmBebidas;

import vista.FrmEntradas;

import vista.FrmFuerte;

import vista.FrmPostre;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

import modelo.Plato;

import modelo.PlatoDAO;

import modelo.Usuario;

import modelo.UsuarioDAO;

import vista.FrmEscogeTuSabor;

import vista.FrmLogin;

import vista.FrmMenu;

import vista.FrmPerfil;

import vista.FrmRegistrar;

import vista.FrmVerIngredientes;

import vista.FrmVersion;

import vista.FrmVisualisarInformacionPedido;

public class ControladorRestaurante implements ActionListener, KeyListener {

Pedido ped= new Pedido();

PedidosDAO pedDAO;

FrmBebidas objetoVistaBebidas;

FrmEntradas objetoVistaEntradas;

FrmFuerte objetoVistaFuerte;

FrmPostre objetoVistaPostre;

FrmConfirmacion objetoVistaConfirmacion;

UsuarioDAO objetoDAO= new UsuarioDAO();

FrmRegistrar objetoRegistrar=new FrmRegistrar();

FrmLogin objetoLogin = new FrmLogin();

FrmPerfil objetoPerfil;

FrmMenu menu = new FrmMenu();

FrmEscogeTuSabor menu2 = new FrmEscogeTuSabor();

Usuario objetoUsuario= new Usuario();

FrmVerIngredientes objetoVistaIngredientes;

FrmVisualisarInformacionPedido objetoVistaFactura;

PlatoDAO platodao;

Plato pla= new Plato();

FrmVersion objetoVersion;

public ControladorRestaurante(

FrmEscogeTuSabor vista,

FrmLogin frmLogin,

FrmRegistrar frmRegistrar,

FrmMenu frmMenu,

UsuarioDAO usuarioDao,

FrmPerfil frmPerfil,

FrmBebidas frmBebidas,

PedidosDAO pedidosDAO,

FrmConfirmacion confirmacion,

FrmVisualisarInformacionPedido factura,

FrmVerIngredientes ingrediente,

PlatoDAO plato,

FrmEntradas frmEntradas,

FrmFuerte frmFuertes,

FrmPostre frmPostre,

FrmVersion frmVersion

){

menu2= vista;

menu2.miRegistrarse.addActionListener(this);

menu2.miLogin.addActionListener(this);

menu2.miSalir.addActionListener(this);

this.objetoLogin = frmLogin;

this.objetoLogin.btnLogin.addActionListener(this);

this.objetoLogin.btnCancelar.addActionListener(this);

this.objetoLogin.txtID.addKeyListener(this);

this.objetoLogin.txtContraseña.addKeyListener(this);

this.objetoRegistrar= frmRegistrar;

this.objetoRegistrar.btnRegistrar.addActionListener(this);

this.objetoRegistrar.btnLogin.addActionListener(this);

this.objetoRegistrar.txtNombre.addKeyListener(this);

this.objetoRegistrar.txtApellido.addKeyListener(this);

this.objetoRegistrar.txtID.addKeyListener(this);

this.objetoRegistrar.txtContraseña.addKeyListener(this);

this.objetoRegistrar.txtRepetirContraseña.addKeyListener(this);

this.objetoRegistrar.txtEmail.addKeyListener(this);

this.objetoRegistrar.txtNumeroCelular.addKeyListener(this);

this.objetoRegistrar.txtDia.addKeyListener(this);

this.objetoRegistrar.txtMes.addKeyListener(this);

this.objetoRegistrar.txtAño.addKeyListener(this);

this.menu = frmMenu;

this.menu.miPerfil.addActionListener(this);

this.menu.btnBebidas.addActionListener(this);

this.menu.btnEntrada.addActionListener(this);

this.menu.btnFuerte.addActionListener(this);

this.menu.btnPostre.addActionListener(this);

this.menu.mHistorial.addActionListener(this);

this.menu.txtNumeroPedido.addKeyListener(this);

this.menu.btnConfirmacionPedidos.addActionListener(this);

this.menu.miCerrarSesion.addActionListener(this);

this.menu.miVersion.addActionListener(this);

this.objetoDAO = usuarioDao;

this.objetoPerfil = frmPerfil;

this.objetoPerfil.btnEditar.addActionListener(this);

this.objetoPerfil.btnModificar.addActionListener(this);

this.objetoPerfil.txtNombre.addKeyListener(this);

this.objetoPerfil.txtApellido.addKeyListener(this);

this.objetoPerfil.txtContraseña.addKeyListener(this);

this.objetoPerfil.txtEmail.addKeyListener(this);

this.objetoPerfil.txtNumeroCelular.addKeyListener(this);

this.objetoPerfil.txtDia.addKeyListener(this);

this.objetoPerfil.txtMes.addKeyListener(this);

this.objetoPerfil.txtAño.addKeyListener(this);

this.objetoPerfil.btnMenu.addActionListener(this);

this.objetoVistaBebidas=frmBebidas;

this.objetoVistaBebidas.rbCoca.addActionListener(this);

this.objetoVistaBebidas.txtCoca.addKeyListener(this);

this.objetoVistaBebidas.rbFanta.addActionListener(this);

this.objetoVistaBebidas.rbSprite.addActionListener(this);

this.objetoVistaBebidas.txtSprite.addActionListener(this);

this.objetoVistaBebidas.txtPedidoBebidas.addKeyListener(this);

this.objetoVistaBebidas.btnAgregarBebidas.addActionListener(this);

this.objetoVistaBebidas.btnRegresar.addActionListener(this);

this.pedDAO = pedidosDAO;

this.objetoVistaConfirmacion = confirmacion;

this.objetoVistaConfirmacion.txtBuscarPedidoconfirmacion.addKeyListener(this);

this.objetoVistaConfirmacion.btnEliminar.addActionListener(this);

this.objetoVistaConfirmacion.btnConfirmar.addActionListener(this);

this.objetoVistaConfirmacion.btnRegresar.addActionListener(this);

this.objetoVistaFactura = factura;

this.objetoVistaFactura.btnVerIngredientes.addActionListener(this);

this.objetoVistaFactura.btnMenu.addActionListener(this);

this.objetoVistaIngredientes = ingrediente ;

this.platodao= plato;

this.objetoVistaIngredientes.txaIngredientes.addKeyListener(this);

this.objetoVistaEntradas = frmEntradas;

this.objetoVistaEntradas.btnAgregarEntrada.addActionListener(this);

this.objetoVistaEntradas.rbBolonVerde.addActionListener(this);

this.objetoVistaEntradas.rbEmpanadaMorocho.addActionListener(this);

this.objetoVistaEntradas.rbPanYuca.addActionListener(this);

this.objetoVistaEntradas.txtCantidadBolon.addKeyListener(this);

this.objetoVistaEntradas.txtCantidadEmpanadasMorocho.addKeyListener(this);

this.objetoVistaEntradas.txtCantidadPan.addKeyListener(this);

this.objetoVistaEntradas.btnRegresarEntrada.addActionListener(this);

this.objetoVistaFuerte=frmFuertes;

this.objetoVistaFuerte.rbArroz.addActionListener(this);

this.objetoVistaFuerte.rbChurrasco.addActionListener(this);

this.objetoVistaFuerte.rbEncebollado.addActionListener(this);

this.objetoVistaFuerte.txtCantidadArroz.addKeyListener(this);

this.objetoVistaFuerte.txtCantidadChurrasco.addKeyListener(this);

this.objetoVistaFuerte.txtCantidadEncebollado.addKeyListener(this);

this.objetoVistaFuerte.btnAgregarFuerte.addActionListener(this);

this.objetoVistaFuerte.jlArrosPrecio.addKeyListener(this);

this.objetoVistaFuerte.jlChurrascoPrecio.addKeyListener(this);

this.objetoVistaFuerte.jlEncebollado.addKeyListener(this);

this.objetoVistaFuerte.btnRegresar.addActionListener(this);

this.objetoVistaPostre=frmPostre;

this.objetoVistaPostre.rbCoco.addActionListener(this);

this.objetoVistaPostre.rbDulceGuayaba.addActionListener(this);

this.objetoVistaPostre.rbEspumillaNaranjilla.addActionListener(this);

this.objetoVistaPostre.btnAgregarPostre.addActionListener(this);

this.objetoVistaPostre.txtCantidadCoco.addKeyListener(this);

this.objetoVistaPostre.txtCantidadDGuayaba.addKeyListener(this);

this.objetoVistaPostre.txtCatidadEspumilla.addKeyListener(this);

this.objetoVistaPostre.txtPedidoPostre.addKeyListener(this);

this.objetoVistaPostre.btnRegresar.addActionListener(this);

this.objetoVersion = frmVersion;

this.objetoVersion.btnMenu.addActionListener(this);

}

public void llenarTabla(JTable tablaD){

DefaultTableModel modeloT= new DefaultTableModel();

tablaD.setModel(modeloT);

modeloT.addColumn("Descripcion ");

modeloT.addColumn("Cantidad");

modeloT.addColumn("Precio");

Object[] columna = new Object[3];

int numReg= pedDAO.obtenerPedidos().size();

for( int i=0; i<numReg;i++){

ped =(Pedido)pedDAO.obtenerPedidos().get(i);

columna[0]= ped.getNombrePedido();

columna[1]= ped.getCantidad();

columna[2]= ped.getPrecio();

modeloT.addRow(columna);

}

}

public void limpiarElementos()

{

objetoPerfil.txtNombre.setText("");

objetoPerfil.txtNombre.setEditable(true);

objetoPerfil.txtApellido.setText("");

objetoPerfil.txtApellido.setEditable(true);

objetoPerfil.txtContraseña.setText("");

objetoPerfil.txtContraseña.setEditable(true);

objetoPerfil.txtEmail.setText("");

objetoPerfil.txtEmail.setEditable(true);

objetoPerfil.txtNumeroCelular.setText("");

objetoPerfil.txtNumeroCelular.setEditable(true);

objetoPerfil.txtMes.setText("");

objetoPerfil.txtMes.setEditable(true);

objetoPerfil.txtAño.setText("");

objetoPerfil.txtAño.setEditable(true);

}

@Override

public void actionPerformed(ActionEvent e) {

if(e.getSource()==menu2.miLogin){

menu2.setVisible(false);

objetoLogin.setVisible(true);

}

if(e.getSource()==menu2.miRegistrarse){

menu2.setVisible(false);

objetoRegistrar.setVisible(true);

}

if(e.getSource()==menu2.miSalir){

this.menu2.dispose();

}

if(e.getSource()==objetoRegistrar.btnRegistrar){

String nombre= objetoRegistrar.txtNombre.getText();

String apellido= objetoRegistrar.txtApellido.getText();

String id= objetoRegistrar.txtID.getText();

String contraseña= objetoRegistrar.txtContraseña.getText();

String email= objetoRegistrar.txtEmail.getText();

String numeroCelular= objetoRegistrar.txtNumeroCelular.getText();

String dia= objetoRegistrar.txtDia.getText();

String mes= objetoRegistrar.txtMes.getText();

String año= objetoRegistrar.txtAño.getText();

Usuario objetoUsuario= new Usuario(nombre,apellido,id,contraseña,email,numeroCelular,dia,mes,año);

objetoDAO.insertarUsuario(objetoUsuario);

}

if(e.getSource()==objetoRegistrar.btnLogin){

objetoRegistrar.setVisible(false);

objetoLogin.setVisible(true);

}

if(e.getSource()== objetoLogin.btnLogin){

String id = "";

String contraseña = "";

id=objetoLogin.txtID.getText();

contraseña=objetoLogin.txtContraseña.getText();

if(objetoDAO.validarLogin(id, contraseña)){

objetoUsuario = objetoDAO.obtenerUsuario(id);

objetoLogin.setVisible(false);

menu.setVisible(true);

}

}

if(e.getSource()==objetoLogin.btnCancelar){

objetoLogin.setVisible(false);

menu2.setVisible(true);

}

if(e.getSource()==objetoVersion.btnMenu){

objetoVersion.setVisible(false);

menu2.setVisible(true);

}

if(e.getSource()==objetoVistaConfirmacion.btnRegresar){

objetoVistaConfirmacion.setVisible(false);

menu.setVisible(true);

}

if(e.getSource()==menu.miPerfil){

menu.setVisible(false);

objetoPerfil.setVisible(true);

objetoPerfil.txtNombre.setText(objetoUsuario.getNombre());

objetoPerfil.txtApellido.setText(objetoUsuario.getApellido());

objetoPerfil.txtID.setText(objetoUsuario.getId());

objetoPerfil.txtContraseña.setText(objetoUsuario.getContraseña());

objetoPerfil.txtEmail.setText(objetoUsuario.getEmail());

objetoPerfil.txtNumeroCelular.setText(objetoUsuario.getNumeroCelular());

objetoPerfil.txtDia.setText(objetoUsuario.getDia());

objetoPerfil.txtMes.setText(objetoUsuario.getMes());

objetoPerfil.txtAño.setText(objetoUsuario.getAño());

objetoPerfil.txtNombre.setEditable(false);

objetoPerfil.txtApellido.setEditable(false);

objetoPerfil.txtID.setEditable(false);

objetoPerfil.txtEmail.setEditable(false);

objetoPerfil.txtContraseña.setEditable(false);

objetoPerfil.txtEmail.setEditable(false);

objetoPerfil.txtNumeroCelular.setEditable(false);

objetoPerfil.txtDia.setEditable(false);

objetoPerfil.txtMes.setEditable(false);

objetoPerfil.txtAño.setEditable(false);

}

if(e.getSource()==objetoPerfil.btnMenu){

objetoPerfil.setVisible(false);

menu.setVisible(true);

}

if(e.getSource()==menu.miVersion){

menu.setVisible(false);

objetoVersion.setVisible(true);

}

if(e.getSource()==menu.miCerrarSesion){

menu.setVisible(false);

menu2.setVisible(true);

}

if(e.getSource()==objetoPerfil.btnModificar){

String nombre = objetoPerfil.txtNombre.getText();

String apellido = objetoPerfil.txtApellido.getText();

String id = objetoPerfil.txtID.getText();

String contraseña = objetoPerfil.txtContraseña.getText();

String email = objetoPerfil.txtEmail.getText();

String numeroCelular = objetoPerfil.txtNumeroCelular.getText();

String dia = objetoPerfil.txtDia.getText();

String mes = objetoPerfil.txtMes.getText();

String año = objetoPerfil.txtAño.getText();

objetoDAO.modificar(nombre, apellido, id, contraseña, email, numeroCelular, dia, mes, año);

objetoPerfil.txtNombre.setEditable(false);

objetoPerfil.txtApellido.setEditable(false);

objetoPerfil.txtID.setEditable(false);

objetoPerfil.txtEmail.setEditable(false);

objetoPerfil.txtContraseña.setEditable(false);

objetoPerfil.txtEmail.setEditable(false);

objetoPerfil.txtNumeroCelular.setEditable(false);

objetoPerfil.txtDia.setEditable(false);

objetoPerfil.txtMes.setEditable(false);

objetoPerfil.txtAño.setEditable(false);

}

if(e.getSource()==objetoPerfil.btnEditar){

objetoPerfil.txtNombre.setEditable(true);

objetoPerfil.txtApellido.setEditable(true);

objetoPerfil.txtEmail.setEditable(true);

objetoPerfil.txtContraseña.setEditable(true);

objetoPerfil.txtEmail.setEditable(true);

objetoPerfil.txtNumeroCelular.setEditable(true);

objetoPerfil.txtDia.setEditable(true);

objetoPerfil.txtMes.setEditable(true);

objetoPerfil.txtAño.setEditable(true);

}

if(e.getSource()==menu.btnConfirmacionPedidos){

menu.setVisible(false);

objetoVistaConfirmacion.setVisible(true);

objetoVistaConfirmacion.txtBuscarPedidoconfirmacion.setText(menu.txtNumeroPedido.getText());

String pedido=objetoVistaConfirmacion.txtBuscarPedidoconfirmacion.getText();

DefaultTableModel modeloT= new DefaultTableModel();

objetoVistaConfirmacion.jlPedidos.setModel(modeloT);

modeloT.addColumn("Descripcion");

modeloT.addColumn("Cantidad");

modeloT.addColumn("Precio");

modeloT.addColumn("Total");

Object [] columna= new Object[4];

int numReg= pedDAO.buscarPedido(pedido).size();

for(int i=0;i<numReg;i++){

ped= (Pedido) pedDAO.buscarPedido(pedido).get(i);

System.out.println(ped);

columna[0]=ped.getNombrePedido();

columna[1]=ped.getCantidad();

columna[2]= ped.getPrecio();

columna[3]= ped.getTotal();

modeloT.addRow(columna);

}

objetoVistaConfirmacion.txtBuscarPedidoconfirmacion.setEditable(false);

}

if(e.getSource()==menu.btnBebidas){

menu.setVisible(false);

objetoVistaBebidas.setVisible(true);

objetoVistaBebidas.txtPedidoBebidas.setText(menu.txtNumeroPedido.getText());

objetoVistaBebidas.txtPedidoBebidas.setEditable(false);

}

if(e.getSource()==menu.btnEntrada){

menu.setVisible(false);

objetoVistaEntradas.setVisible(true);

objetoVistaEntradas.txtPedidoEntrada.setText(menu.txtNumeroPedido.getText());

objetoVistaEntradas.txtPedidoEntrada.setEditable(false);

}

if(e.getSource()==menu.btnFuerte){

menu.setVisible(false);

objetoVistaFuerte.setVisible(true);

objetoVistaFuerte.txtPedidoFuertes.setText(menu.txtNumeroPedido.getText());

objetoVistaFuerte.txtPedidoFuertes.setEditable(false);

}

if(e.getSource()==menu.btnPostre){

menu.setVisible(false);

objetoVistaPostre.setVisible(true);

objetoVistaPostre.txtPedidoPostre.setText(menu.txtNumeroPedido.getText());

objetoVistaPostre.txtPedidoPostre.setEditable(false);

}

if(e.getSource()==objetoVistaBebidas.btnAgregarBebidas){

String numPedido=objetoVistaBebidas.txtPedidoBebidas.getText();

if(objetoVistaBebidas.rbCoca.getLabel().equalsIgnoreCase("Coca cola")){

String nombrePed=objetoVistaBebidas.rbCoca.getLabel();

String cantidad= objetoVistaBebidas.txtCoca.getText();

String precio= objetoVistaBebidas.jlCoca.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(objetoVistaBebidas.jlCoca.getText());

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

if(objetoVistaBebidas.rbFanta.getLabel().equalsIgnoreCase("Fanta")){

String nombrePed=objetoVistaBebidas.rbFanta.getLabel();

String cantidad= objetoVistaBebidas.txtFanta.getText();

String precio= objetoVistaBebidas.jlFanta.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(objetoVistaBebidas.jlFanta.getText());

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

if(objetoVistaBebidas.rbSprite.getLabel().equalsIgnoreCase("Sprite")){

String nombrePed=objetoVistaBebidas.rbSprite.getLabel();

String cantidad= objetoVistaBebidas.txtSprite.getText();

String precio=objetoVistaBebidas.jlSprite.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(objetoVistaBebidas.jlSprite.getText());

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

}

if(e.getSource()==objetoVistaBebidas.btnRegresar){

objetoVistaBebidas.setVisible(false);

menu.setVisible(true);

}

if(e.getSource()==objetoVistaPostre.btnAgregarPostre){

String numPedido=objetoVistaPostre.txtPedidoPostre.getText();

if(objetoVistaPostre.rbDulceGuayaba.getLabel().equalsIgnoreCase("Dulce de Guayaba")){

String nombrePed=objetoVistaPostre.rbDulceGuayaba.getLabel();

String cantidad= objetoVistaPostre.txtCantidadDGuayaba.getText();

String precio=objetoVistaPostre.jlPrecioGuayaba.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(precio);

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

if(objetoVistaPostre.rbEspumillaNaranjilla.getLabel().equalsIgnoreCase("Espumilla de Naranjilla")){

String nombrePed=objetoVistaPostre.rbEspumillaNaranjilla.getLabel();

String cantidad= objetoVistaPostre.txtCatidadEspumilla.getText();

String precio=objetoVistaPostre.jlPrecioEspumilla.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(precio);

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

if(objetoVistaPostre.rbCoco.getLabel().equalsIgnoreCase("Bien me sabe de Coco")){

String nombrePed=objetoVistaPostre.rbCoco.getLabel();

String cantidad= objetoVistaPostre.txtCantidadCoco.getText();

String precio=objetoVistaPostre.jlPrecioBienCoco.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(precio);

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

}

if(e.getSource()==objetoVistaPostre.btnRegresar){

objetoVistaPostre.setVisible(false);

menu.setVisible(true);

}

if(e.getSource()==objetoVistaFuerte.btnAgregarFuerte){

String numPedido=objetoVistaFuerte.txtPedidoFuertes.getText();

if(objetoVistaFuerte.rbArroz.getLabel().equalsIgnoreCase("Arroz con Conchas")){

String nombrePed=objetoVistaFuerte.rbArroz.getLabel();

String cantidad= objetoVistaFuerte.txtCantidadArroz.getText();

String precio=objetoVistaFuerte.jlArrosPrecio.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(objetoVistaFuerte.jlArrosPrecio.getText());

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

if(objetoVistaFuerte.rbChurrasco.getLabel().equalsIgnoreCase("Churrasco")){

String nombrePed=objetoVistaFuerte.rbChurrasco.getLabel();

String cantidad= objetoVistaFuerte.txtCantidadChurrasco.getText();

String precio=objetoVistaFuerte.jlChurrascoPrecio.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(objetoVistaFuerte.jlChurrascoPrecio.getText());

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

if(objetoVistaFuerte.rbEncebollado.getLabel().equalsIgnoreCase("Encebollado")){

String nombrePed=objetoVistaFuerte.rbEncebollado.getLabel();

String cantidad= objetoVistaFuerte.txtCantidadEncebollado.getText();

String precio=objetoVistaFuerte.jlEncebollado.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(objetoVistaFuerte.jlEncebollado.getText());

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

}

if(e.getSource()==objetoVistaFuerte.btnRegresar){

objetoVistaFuerte.setVisible(false);

menu.setVisible(true);

}

if(e.getSource()==objetoVistaEntradas.btnAgregarEntrada){

String numPedido=objetoVistaEntradas.txtPedidoEntrada.getText();

if(objetoVistaEntradas.rbBolonVerde.getLabel().equalsIgnoreCase("Bolon de Verde")){

String nombrePed=objetoVistaEntradas.rbBolonVerde.getLabel();

String cantidad= objetoVistaEntradas.txtCantidadBolon.getText();

String precio=objetoVistaEntradas.jlPrecioBolon.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(objetoVistaEntradas.jlPrecioBolon.getText());

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

if(objetoVistaEntradas.rbEmpanadaMorocho.getLabel().equalsIgnoreCase("Empanadas de morocho")){

String nombrePed=objetoVistaEntradas.rbEmpanadaMorocho.getLabel();

String cantidad= objetoVistaEntradas.txtCantidadEmpanadasMorocho.getText();

String precio=objetoVistaEntradas.jlPrecioEmpanadas.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(objetoVistaEntradas.jlPrecioEmpanadas.getText());

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

if(objetoVistaEntradas.rbPanYuca.getLabel().equalsIgnoreCase("Pan de Yuca")){

String nombrePed=objetoVistaEntradas.rbPanYuca.getLabel();

String cantidad= objetoVistaEntradas.txtCantidadPan.getText();

String precio=objetoVistaEntradas.jlPanYuca.getText();

float total = Integer.parseInt(cantidad)\*Float.parseFloat(objetoVistaEntradas.jlPanYuca.getText());

Pedido objPed= new Pedido(numPedido,nombrePed, Integer.parseInt(cantidad),precio,valueOf(total));

pedDAO.insertarPedidos(objPed);

}

}

if(e.getSource()==objetoVistaEntradas.btnRegresarEntrada){

objetoVistaEntradas.setVisible(false);

menu.setVisible(true);

}

if(e.getSource()==objetoVistaConfirmacion.btnEliminar){

int filaInicio=objetoVistaConfirmacion.jlPedidos.getSelectedRow();

int numFS= objetoVistaConfirmacion.jlPedidos.getSelectedRowCount();

ArrayList<String> lista= new ArrayList();

String nom="";

if(filaInicio>0){

for(int i=0;1<numFS;i++){

nom=String.valueOf(objetoVistaConfirmacion.jlPedidos.getValueAt(i+filaInicio,0 ));

lista.add(nom);

}

for(int i=0;i<numFS;i++){

int respuesta= JOptionPane.showConfirmDialog(null,"Quiere eliminar el pedido:\n"+nom+"?");

if (respuesta==0){

pedDAO.eliminarPedidos(nom);

}

llenarTabla(objetoVistaConfirmacion.jlPedidos);

}

}else{

JOptionPane.showMessageDialog(null, "Debe seleccionar una fila a eliminar");

}

}

if(e.getSource()==objetoVistaConfirmacion.btnConfirmar){

float total=0;

String pedido=objetoVistaConfirmacion.txtBuscarPedidoconfirmacion.getText();

objetoVistaFactura.nombreCliente.setText(objetoVistaConfirmacion.txtNombreCliente.getText());

objetoVistaFactura.numeroPedido.setText(objetoVistaConfirmacion.txtBuscarPedidoconfirmacion.getText());

DefaultTableModel modeloT= new DefaultTableModel();

objetoVistaFactura.jtFactura.setModel(modeloT);

modeloT.addColumn("Descripcion");

modeloT.addColumn("Cantidad");

modeloT.addColumn("Precio");

modeloT.addColumn("Total");

Object [] columna= new Object[4];

int numReg= pedDAO.buscarPedido(pedido).size();

for(int i=0;i<numReg;i++){

ped= (Pedido) pedDAO.buscarPedido(pedido).get(i);

System.out.println(ped);

columna[0]=ped.getNombrePedido();

columna[1]=ped.getCantidad();

columna[2]= ped.getPrecio();

columna[3]= ped.getTotal();

total = total + Float.parseFloat(ped.getTotal());

modeloT.addRow(columna);

}

objetoVistaFactura.labelTotal.setText(Float.toString(total));

objetoVistaConfirmacion.setVisible(false);

objetoVistaFactura.setVisible(true);

}

if(e.getSource()==objetoVistaFactura.btnVerIngredientes){

String pedido=objetoVistaConfirmacion.txtBuscarPedidoconfirmacion.getText();

objetoVistaIngredientes.LabelCliente.setText(objetoVistaFactura.nombreCliente.getText());

objetoVistaIngredientes.LabelNumeroPedido.setText(objetoVistaFactura.numeroPedido.getText());

int numReg= pedDAO.buscarPedido(pedido).size();

for(int i=0;i<numReg;i++){

ped= (Pedido) pedDAO.buscarPedido(pedido).get(i);

System.out.println(ped);

int numRe= platodao.buscarPlato(ped.getNombrePedido()).size();

for(int j=0;i<numRe;j++){

pla= (Plato) platodao.buscarPlato(ped.getNombrePedido()).get(i);

objetoVistaIngredientes.txaIngredientes.setText(pla.getNombre());

objetoVistaIngredientes.txaIngredientes.setEditable(false);

// String pedido= objetoVistaVisualisar.txtOrden.getText();

// objetoVistaVering= new FrmVerIngredientes();

// objetoVistaVering.LabelCliente.setText(objetoVistaVisualisar.txtNombreF.getText());

// objetoVistaVering.LabelNumPedido.setText(objetoVistaVisualisar.txtOrden.getText());

// objetoVistaVisualisar.setVisible(false);

// DefaultTableModel modeloT= new DefaultTableModel();

// objetoVistaVering.jtVerIn.setModel(modeloT);

// modeloT.addColumn("Nombre del plato");

// modeloT.addColumn("Ingrediente");

// modeloT.addColumn("Cantidad");

// modeloT.addColumn("Unidades");

// modeloT.addColumn("total de ingredientes");

// Object [] columna= new Object[5];

// int numReg= pedDAO.buscarPedido(pedido).size();

// for(int i=0;i<numReg;i++){

// ped= (Pedido) pedDAO.buscarPedido(pedido).get(i);

//// System.out.println(ped);

//// int numRe= platoDAO.buscarPlato(ped.getNombrePedido()).size();

//// for(int j=0;i<numRe;j++){

//// plato= (Plato) platoDAO.buscarPlato(ped.getNombrePedido()).get(i);

//// columna[0]= plato.getNombre();

//// columna[1]=plato.getIngredientes();

//// columna[2]=ped.getCantidad();

////// columna[1]=;

////// columna[1]=;

////// ped.getNombrePedido() = " ";

//// }

// modeloT.addRow(columna);

// }

// objetoVistaVering.setVisible(true);

}

objetoVistaIngredientes.setVisible(true);

objetoVistaFactura.setVisible(false);

}

}

if(e.getSource()==objetoVistaFactura.btnMenu){

objetoVistaFactura.setVisible(false);

menu.setVisible(true);

}

}

@Override

public void keyTyped(KeyEvent e) {

if (e.getSource()== objetoRegistrar.txtNombre)

{

char c= e.getKeyChar();

if ((c<'a' || c>'z') && (c< 'A'|| c >'Z'))

{

e.consume();

}

}

if (e.getSource()== objetoRegistrar.txtApellido)

{

char c= e.getKeyChar();

if ((c<'a' || c>'z') && (c< 'A'|| c >'Z'))

{

e.consume();

}

}

if (e.getSource()== objetoRegistrar.txtID)

{

char c= e.getKeyChar();

if ((c<'0' || c>'9') )

{

e.consume();

}

}

if (e.getSource()== objetoRegistrar.txtNumeroCelular)

{

char c= e.getKeyChar();

if ((c<'0' || c>'9') )

{

e.consume();

}

}

if (e.getSource()== objetoPerfil.txtNombre)

{

char c= e.getKeyChar();

if ((c<'a' || c>'z') && (c< 'A'|| c >'Z'))

{

e.consume();

}

}

if (e.getSource()== objetoPerfil.txtApellido)

{

char c= e.getKeyChar();

if ((c<'a' || c>'z') && (c< 'A'|| c >'Z'))

{

e.consume();

}

}

if (e.getSource()== objetoPerfil.txtNumeroCelular)

{

char c= e.getKeyChar();

if ((c<'0' || c>'9') )

{

e.consume();

}

}

}

@Override

public void keyPressed(KeyEvent e) {

String id= objetoRegistrar.txtID.getText();

int lenght1 = id.length();

if(e.getKeyChar()>='0' && e.getKeyChar()<='9'){

if (lenght1<10){

objetoRegistrar.txtID.setEditable(true);

}else{

objetoRegistrar.txtID.setEditable(false);

}

}

String numeroCelular1= objetoRegistrar.txtNumeroCelular.getText();

int lenght2 = numeroCelular1.length();

if(e.getKeyChar()>='0' && e.getKeyChar()<='9'){

if (lenght2<10){

objetoRegistrar.txtNumeroCelular.setEditable(true);

}else{

objetoRegistrar.txtNumeroCelular.setEditable(false);

}

}

String numeroCelular2= objetoPerfil.txtNumeroCelular.getText();

int lenght3 = id.length();

if(e.getKeyChar()>='0' && e.getKeyChar()<='9'){

if (lenght3<10){

objetoPerfil.txtNumeroCelular.setEditable(true);

}else{

objetoPerfil.txtNumeroCelular.setEditable(false);

}

}

String id2= objetoRegistrar.txtID.getText();

int lenght4 = id.length();

if(e.getKeyChar()>='0' && e.getKeyChar()<='9'){

if (lenght1<10){

objetoRegistrar.txtID.setEditable(true);

}else{

objetoRegistrar.txtID.setEditable(false);

}

}

}

@Override

public void keyReleased(KeyEvent e) {

}

}

**Gestor Restaurante**

package controlador;

import modelo.Conexion;

import modelo.Ingrediente;

import modelo.PedidosDAO;

import modelo.Plato;

import vista.FrmBebidas;

import vista.FrmConfirmacion;

import modelo.UsuarioDAO;

import vista.FrmBebidas;

import vista.FrmConfirmacion;

import vista.FrmRegistrar;

import modelo.Ingrediente;

import modelo.Plato;

import vista.FrmBebidas;

import vista.FrmConfirmacion;

import modelo.Ingrediente;

import modelo.PlatoDAO;

import vista.FrmBebidas;

import vista.FrmConfirmacion;

import vista.FrmEntradas;

import vista.FrmEscogeTuSabor;

import vista.FrmFuerte;

import vista.FrmLogin;

import vista.FrmMenu;

import vista.FrmPerfil;

import vista.FrmPostre;

import vista.FrmVerIngredientes;

import vista.FrmVersion;

import vista.FrmVisualisarInformacionPedido;

public class GestorRestaurante {

//Camilo

public static void main(String args[]){

FrmEscogeTuSabor menuPrimario= new FrmEscogeTuSabor();

FrmRegistrar r = new FrmRegistrar();

FrmLogin login = new FrmLogin();

FrmMenu menuSecundario = new FrmMenu();

UsuarioDAO dao = new UsuarioDAO();

FrmPerfil perfil = new FrmPerfil();

FrmBebidas bebidas = new FrmBebidas();

PedidosDAO pedDAO = new PedidosDAO();

FrmEntradas entradas = new FrmEntradas();

FrmConfirmacion frmConfirmar = new FrmConfirmacion();

FrmVisualisarInformacionPedido factura = new FrmVisualisarInformacionPedido();

FrmVerIngredientes ingrediente = new FrmVerIngredientes ();

PlatoDAO plato = new PlatoDAO ();

FrmFuerte frmFuerte= new FrmFuerte();

FrmPostre frmPostre= new FrmPostre();

FrmVersion frmVersion= new FrmVersion();

ControladorRestaurante c = new ControladorRestaurante(

menuPrimario,

login,

r,

menuSecundario,

dao,

perfil,

bebidas,

pedDAO,

frmConfirmar,

factura,

ingrediente,

plato,

entradas,

frmFuerte,

frmPostre,

frmVersion

);

menuPrimario.setVisible(true);

};

}

**Modelo**

**CLASE Conexión**

package modelo;

import com.mongodb.DB;

import com.mongodb.DBCollection;

import com.mongodb.MongoClient;

public class Conexion {

DB baseDatos;

DBCollection coleccion;

DBCollection coleccionPedido;

DBCollection coleccionPlato;

DBCollection coleccionIngre;

//Constructor

public Conexion(){

MongoClient mongo=new MongoClient("localhost",27017);

baseDatos= mongo.getDB("Restaurante");

coleccion= baseDatos.getCollection("Usuario");

coleccionPedido=baseDatos.getCollection("Pedidos");

coleccionPlato=baseDatos.getCollection("Plato");

coleccionIngre=baseDatos.getCollection("Ingrediente");

System.out.println("Conexion aBD ok");

}

}

Ingredientes

package modelo;

import java.util.Iterator;

public class Ingrediente {

private String nombre;

private int cantidad;

public Ingrediente(String nombre, int cantidad) {

this.nombre = nombre;

this.cantidad = cantidad;

}

public Ingrediente() {

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public int getCantidad() {

return cantidad;

}

public void setCantidad(int cantidad) {

this.cantidad = cantidad;

}

@Override

public String toString() {

return "Ingrediente{" + "nombre=" + nombre + ", cantidad=" + cantidad + '}';

}

}

**CLASE Pedidos**

package modelo;

public class Pedido {

public String numeroPedido;

public String nombrePedido;// nombre del plato

public int cantidad;

public String precio1;

public String total;

public Pedido(String numeroPedido, String nombrePedido, int cantidad,String precio1, String total) {

this.numeroPedido = numeroPedido;

this.nombrePedido = nombrePedido;

this.cantidad = cantidad;

this.precio1=precio1;

this.total=total;

}

public Pedido() {

}

public void setPrecio1(String precio1) {

this.precio1 = precio1;

}

public void setTotal(String total) {

this.total = total;

}

public String getPrecio1() {

return precio1;

}

public String getTotal() {

return total;

}

public String getPrecio() {

return precio1;

}

public String getNumeroPedido() {

return numeroPedido;

}

public String getNombrePedido() {

return nombrePedido;

}

public int getCantidad() {

return cantidad;

}

public void setNombrePedido(String nombrePedido) {

this.nombrePedido = nombrePedido;

}

public void setNumeroPedido(String numeroPedido) {

this.numeroPedido = numeroPedido;

}

public void setCantidad(int cantidad) {

this.cantidad = cantidad;

}

public void setPrecio(String precio) {

this.precio1 = precio;

}

@Override

public String toString() {

return "Pedido{" + "numeroPedido=" + numeroPedido + ", nombrePedido=" + nombrePedido + ", cantidad=" + cantidad + ", precio1=" + precio1 + ", total=" + total + '}';

}

}

**CLASE PedidosDAO**

package modelo;

import com.mongodb.BasicDBObject;

import com.mongodb.DBCursor;

import java.util.ArrayList;

public class PedidosDAO {

public void insertarPedidos(Pedido ped){

Conexion objCon = new Conexion();

BasicDBObject documento = new BasicDBObject();

documento.put("npedido",ped.getNumeroPedido());

documento.put("nombrePedido",ped.getNombrePedido());

documento.put("cantidad",ped.getCantidad());

documento.put("precio",ped.getPrecio());

documento.put("total",ped.getTotal());

objCon.coleccionPedido.insert(documento);

}

public void eliminarPedidos(String nombrePedido){

Conexion objCon = new Conexion();

BasicDBObject documento = new BasicDBObject();

documento.put("nombrePedido", nombrePedido);

objCon.coleccionPedido.remove(documento);

}

public ArrayList obtenerPedidos(){

ArrayList<Pedido> listaPedidos= new ArrayList<Pedido>();

Pedido aux1= null;

Conexion objCon= new Conexion();

DBCursor cursor= objCon.coleccionPedido.find();

while (cursor.hasNext()){

aux1= new Pedido((String) cursor.next().get("npedido"),

(String) cursor.next().get("nombrePedido"),

(int) cursor.next().get("cantidad"),(String) cursor.next().get("precio"),(String) cursor.next().get("total"));

listaPedidos.add(aux1);

}

return listaPedidos;

}

public ArrayList buscarPedido(String pedidos) {

ArrayList<Pedido> listPe = new ArrayList();

Pedido p1;

Conexion objCon = new Conexion();

BasicDBObject buscado = new BasicDBObject ("npedido", pedidos);

DBCursor cursor = objCon.coleccionPedido.find(buscado);

while(cursor.hasNext()){

//primero se utiliza el next y luego los curr

p1 = new Pedido((String)cursor.next().get("npedido"),

(String)cursor.curr().get("nombrePedido"),

(int)cursor.curr().get("cantidad"),(String)cursor.curr().get("precio"),(String)cursor.curr().get("total"));

listPe.add(p1);

}

return listPe;

}

}

**CLASE Plato**

package modelo;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.List;

public class Plato {

private String nombre;

List ingredientes;

public Plato(String nombre, List ingredientes) {

this.nombre = nombre;

this.ingredientes = ingredientes;

}

public Plato() {

}

public Plato(String nombre) {

this.nombre = nombre;

ingredientes = new ArrayList<Ingrediente>();

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public List getIngredientes() {

return ingredientes;

}

public void setIngredientes(List ingredientes) {

this.ingredientes = ingredientes;

}

public void addIn(Ingrediente obj){

ingredientes.add(obj);

}

public int tamanio(){

return ingredientes.size();

}

public float calcularCantidad(){

float res=0;

Ingrediente aux =new Ingrediente ();

Iterator objIterator= ingredientes.iterator();

while (objIterator.hasNext()){

aux=(Ingrediente)objIterator.next();//Don casting debo espesificar de dnde viene

res= res + aux.getCantidad();

}

//res=

return res;

}

@Override

public String toString() {

return "Plato{" + "nombre=" + nombre + ", ingredientes=" + ingredientes + '}';

}

public void mostar(){

float res=0;

Ingrediente aux =new Ingrediente ();

Iterator objIterator= ingredientes.iterator();

while (objIterator.hasNext()){

aux=(Ingrediente)objIterator.next();//Don casting debo espesificar de dnde viene

System.out.println(aux.toString());

}

}

}

**CLASE PlatoDAO**

package modelo;

import com.mongodb.BasicDBObject;

import com.mongodb.DBCursor;

import java.util.ArrayList;

import java.util.\*;

public class PlatoDAO {

public ArrayList obtenerPlato(){

ArrayList<Plato> listaPlato= new ArrayList<Plato>();

Plato aux1= null;

Conexion objCon= new Conexion();

DBCursor cursor= objCon.coleccionPedido.find();

while (cursor.hasNext()){

aux1= new Plato((String) cursor.next().get("nombre"),

(ArrayList)cursor.next().get("Ingredientes"));

listaPlato.add(aux1);

}

return listaPlato;

}

public ArrayList buscarPlato(String nombre) {

ArrayList<Plato> listPe = new ArrayList();

Plato p1;

Conexion objCon = new Conexion();

BasicDBObject buscado = new BasicDBObject ("nombre", nombre);

DBCursor cursor = objCon.coleccionPedido.find(buscado);

while(cursor.hasNext()){

//primero se utiliza el next y luego los curr

p1 = new Plato((String)cursor.next().get("nombre"),

(List)cursor.curr().get("Ingredientes"));

listPe.add(p1);

listPe.toString();

}

return listPe;

}

}

**CLASE Usuario**

package modelo;

public class Usuario {

private String nombre;

private String apellido;

private String id;

private String contraseña;

private String email;

private String numeroCelular;

private String dia;

private String mes;

private String año;

public Usuario() {

}

public Usuario(String id, String contraseña) {

this.id = id;

this.contraseña = contraseña;

}

public Usuario(String nombre, String apellido, String id, String contraseña, String email, String numeroCelular, String dia, String mes, String año) {

this.nombre = nombre;

this.apellido = apellido;

this.id = id;

this.contraseña = contraseña;

this.email = email;

this.numeroCelular = numeroCelular;

this.dia = dia;

this.mes = mes;

this.año = año;

}

public String getNombre() {

return nombre;

}

public String getApellido() {

return apellido;

}

public String getId() {

return id;

}

public String getContraseña() {

return contraseña;

}

public String getEmail() {

return email;

}

public String getNumeroCelular() {

return numeroCelular;

}

public String getDia() {

return dia;

}

public String getMes() {

return mes;

}

public String getAño() {

return año;

}

}

**CLASE UsuarioDAO**

package modelo;

import com.mongodb.BasicDBObject;

import com.mongodb.DBCursor;

import com.mongodb.DBObject;

public class UsuarioDAO {

private Conexion objCon;

public UsuarioDAO() {

objCon=new Conexion();

}

public void insertarUsuario(Usuario p){

BasicDBObject documento= new BasicDBObject();

documento.put("nombre", p.getNombre());

documento.put("apellido", p.getApellido());

documento.put("id", p.getId());

documento.put("contraseña", p.getContraseña());

documento.put("email", p.getEmail());

documento.put("numeroCelular", p.getNumeroCelular());

documento.put("dia", p.getDia());

documento.put("mes", p.getMes());

documento.put("año", p.getAño());

objCon.coleccion.insert(documento);

}

public Usuario obtenerUsuario(String id) {

DBCursor cursor = objCon.coleccion.find();

while (cursor.hasNext()){

DBObject dbObject = cursor.next();

if (((String)dbObject.get("id")).equals(id)){

return new Usuario(

(String)dbObject.get("nombre"),

(String)dbObject.get("apellido"),

(String)dbObject.get("id"),

(String)dbObject.get("contraseña"),

(String)dbObject.get("email"),

(String)dbObject.get("numeroCelular"),

(String)dbObject.get("dia"),

(String)dbObject.get("mes"),

(String)dbObject.get("año")

);

}

}

throw new IllegalArgumentException();

}

public void modificar(String nombre1,

String apellido1,

String id1,

String contraseña1,

String email1,

String numeroCelular1,

String dia1,

String mes1,

String anio1){

DBObject buscado= new BasicDBObject("id",id1);

DBObject nombre= new BasicDBObject().append("$set",new BasicDBObject().append("nombre",nombre1));

DBObject apellido= new BasicDBObject().append("$set",new BasicDBObject().append("apellido",apellido1));

DBObject contraseñaNuevo= new BasicDBObject().append("$set",new BasicDBObject().append("contraseña",contraseña1));

DBObject correo= new BasicDBObject().append("$set",new BasicDBObject().append("email",email1));

DBObject nCelular= new BasicDBObject().append("$set",new BasicDBObject().append("numeroCelular",numeroCelular1));

DBObject dia= new BasicDBObject().append("$set",new BasicDBObject().append("dia",dia1));

DBObject mes= new BasicDBObject().append("$set",new BasicDBObject().append("mes",mes1));

DBObject anio= new BasicDBObject().append("$set",new BasicDBObject().append("año",anio1));

objCon.coleccion.update(buscado, nombre);

objCon.coleccion.update(buscado, apellido);

objCon.coleccion.update(buscado, contraseñaNuevo);

objCon.coleccion.update(buscado, correo);

objCon.coleccion.update(buscado, nCelular);

objCon.coleccion.update(buscado, dia);

objCon.coleccion.update(buscado, mes);

objCon.coleccion.update(buscado, anio);

// DBObject nombreNuevo = new BasicDBObject ("nombre", nombre);

// DBObject apellidoNuevo = new BasicDBObject ("apellido", apellido);

// DBObject nuevo= new BasicDBObject().append("$set", new BasicDBObject().append("id", id));

// DBObject contraseñaNuevo = new BasicDBObject ("contraseña", contraseña);

// DBObject emailNuevo = new BasicDBObject ("email", email);

// DBObject numeroCelularNuevo = new BasicDBObject ("numeroCelular", numeroCelular);

// DBObject diaNuevo = new BasicDBObject ("dia", dia);

// DBObject mesNuevo = new BasicDBObject ("mes", mes);

// DBObject añoNuevo = new BasicDBObject ("año", año);

// objCon.coleccion.update(nombreNuevo,nuevo);

// objCon.coleccion.update(apellidoNuevo, nuevo);

// objCon.coleccion.update(contraseñaNuevo, nuevo);

// objCon.coleccion.update(emailNuevo, nuevo);

// objCon.coleccion.update(numeroCelularNuevo, nuevo);

// objCon.coleccion.update(diaNuevo, nuevo);

// objCon.coleccion.update(mesNuevo, nuevo);

// objCon.coleccion.update(buscado, nuevo);

}

public boolean validarLogin(String id,String contraseña){

Usuario aux = new Usuario();

DBCursor cursor = objCon.coleccion.find();

boolean abrir= false;

while (cursor.hasNext()){

aux = new Usuario((String) cursor.next().get("id"),(String)cursor.curr().get("contraseña"));

if (aux.getId().equals(id) && aux.getContraseña().equals(contraseña)){

abrir = true;

}

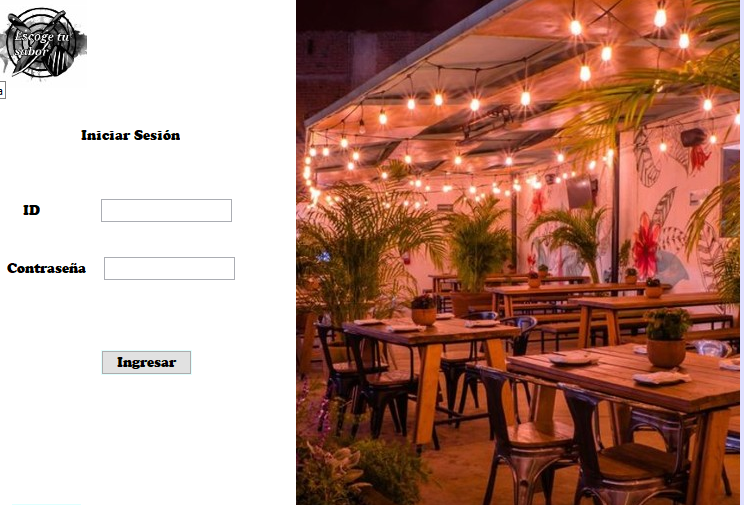
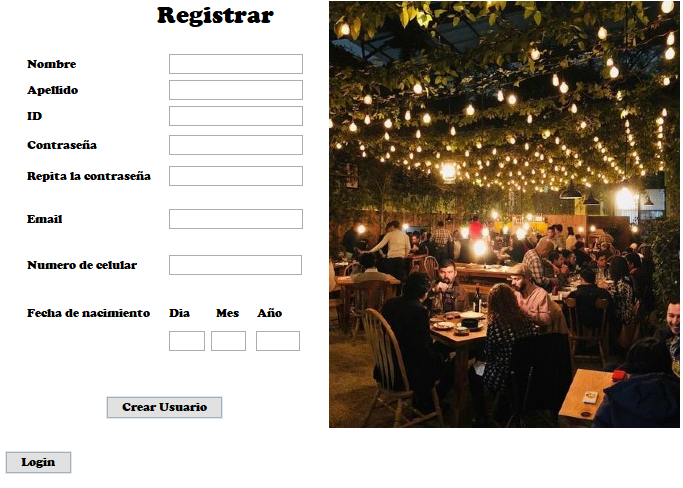
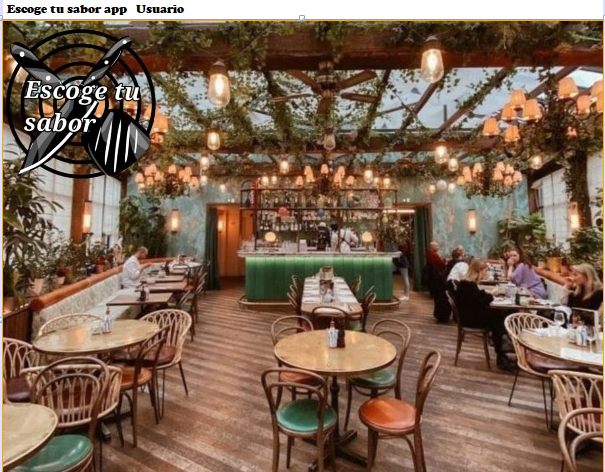
}

return abrir;

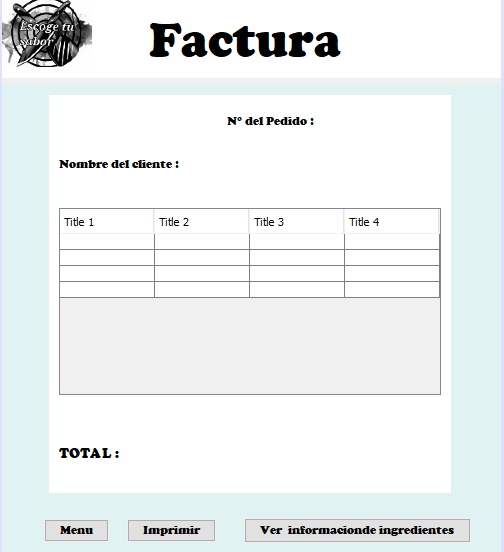
}

}

# 6) Captura de la interfaz grafica







# 7 captura de la pantalla de la creación de datos



8 captura de pantallas corridas 