

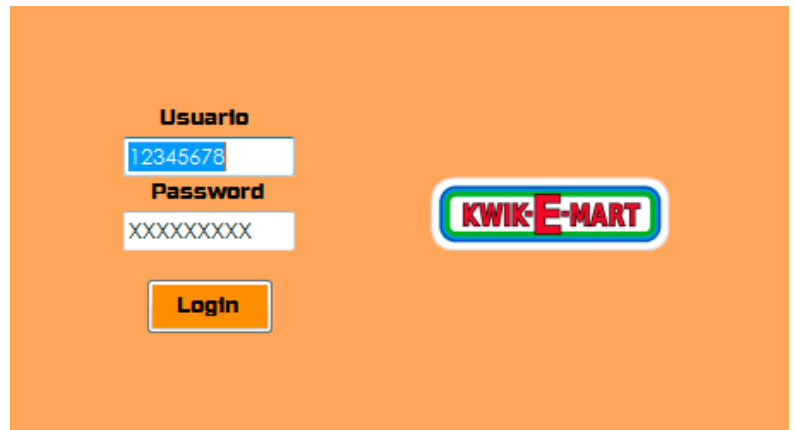
Trabajo practico 4 Laboratorio II

Alvaro Villegas 2D

Aplicación de Ventas

Login

Compara el número del Usuario y la contraseña, con la base de Datos



The login screen features an orange background. On the left, there are two input fields: the top one is labeled 'Usuario' and contains the text '12345678', while the bottom one is labeled 'Password' and contains 'XXXXXXXX'. Below these fields is an orange button with the text 'Login'. To the right of the input fields is the 'KWIK-E-MART' logo, which consists of the brand name in red capital letters inside a green-bordered rounded rectangle.

Menu

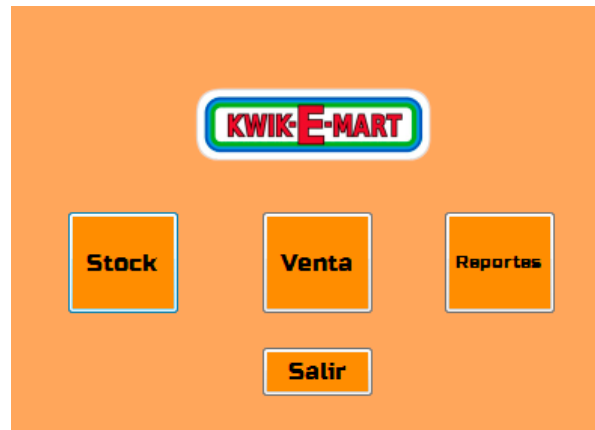
Menu Estandar

Stock : Controla y Añade Productos

Venta: Para hacer Ventas

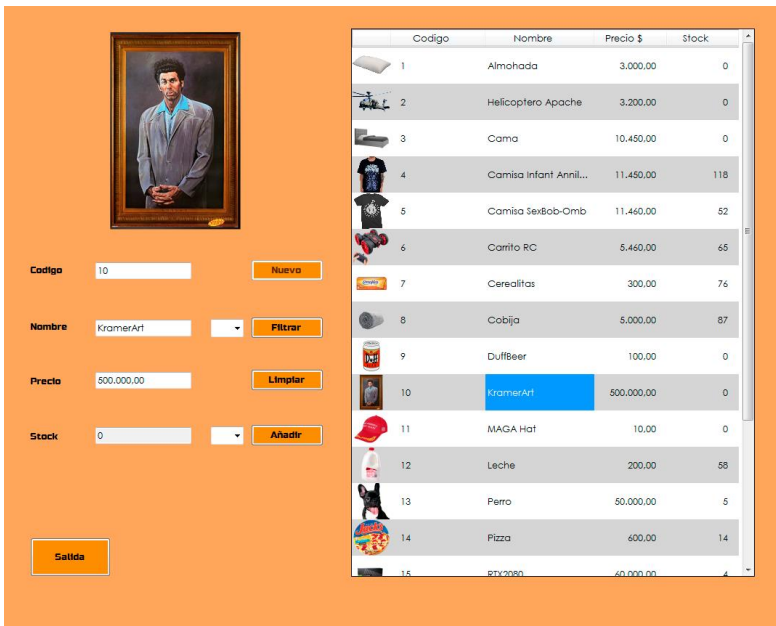
Reportes: Para ver Ventas realizadas

Salir : Volver al Login



The main menu screen has an orange background. At the top center is the 'KWIK-E-MART' logo. Below the logo, there are four orange buttons arranged in a grid: 'Stock' on the left, 'Venta' in the center, 'Reportes' on the right, and 'Salir' at the bottom center.

Stock



Código: 10 **Nuevo**

Nombre: KramerArt **Filtrar**

Precio: 500.000,00 **Limpiar**

Stock: 0 **Añadir**

Salida

Código	Nombre	Precio \$	Stock
1	Almohada	3.000,00	0
2	Helicoptero Apache	3.200,00	0
3	Cama	10.450,00	0
4	Camisa Infant Annil...	11.450,00	118
5	Camisa Sex&Bob-Omb	11.460,00	52
6	Carrito RC	5.460,00	65
7	Cerealitas	300,00	76
8	Cabija	5.000,00	87
9	DuffBeer	100,00	0
10	KramerArt	500.000,00	0
11	MAGA Hat	10,00	0
12	Leche	200,00	58
13	Perro	50.000,00	5
14	Pizza	600,00	14
15	RTV2080	60.000,00	4

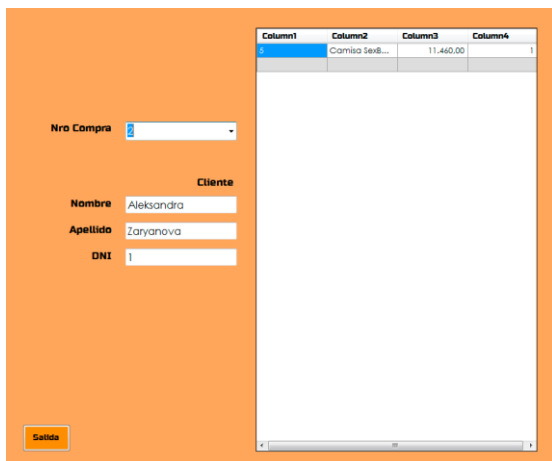
Se carga Automáticamente los Productos y sus datos.

Si selecciona un Producto en la lista o se ingresa el Código en el campo **“Codigo”**, muestra el producto correspondiente.

Para Añadir Cantidades al Stock de cada producto, Seleccionar un Producto y seleccionar o ingresar una cantidad en añadir; y apretar el botón **“Cantidad”**.

Para Añadir un producto: Ingresar un nombre y un precio y apretar el botón **“Nuevo”**.

Reportes



Nro Compra: 2

Cliente:

Nombre: Aleksandra

Apellido: Zaryanova

DNI: 1

Salida

Column1	Column2	Column3	Column4
	Camisa Sex&...	11.460,00	1

Seleccionar alguna compra en el desplegable **“Nro Compra”** y seleccionar alguna compra previamente cargada, para mostrar los datos del Cliente y Productos comprados

Ventas

DNI 1 **Nombre** Aleksandra **Apellido** Zaryanova

Codigo	Producto	Precio	Cantidad
4	Camisa Infant Annih...	11.450,00	1

SubTotal: 11.450,00
Descuento : 0
Total: 11.450,00

Id **Nombre** **Precio** **Stock**

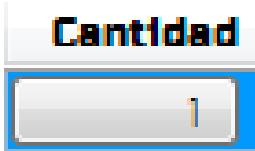
1	Almohada	3000	0
2	Helicoptero Ap...	3200	0
3	Cama	10450	0
4	Camisa Infant ...	11450	118
5	Camisa SexBob-...	11460	52
6	Carrito RC	5460	65
7	Cerealitas	300	76
8	Cobija	5000	87
9	DuffBeer	100	0
10	KramerArt	500000	0
11	MAGA Hat	10	0
12	Leche	200	58
13	Perro	50000	5
14	Pizza	600	14

Salir **Venta**

25

Se ingresa un DNI : si ya existe, trae los datos automáticamente, si es un DNI nuevo se crea y guarda el Cliente al cierre de la venta.

Arrastre Productos de la lista para ingresarlos a la compra (Productos que tengan Stock positivo).



Para modificar la cantidad del Producto a vender, arrastrar otra vez o modificar en el botón de cantidad.



Para quitar un producto de la Venta, click en la “X”.

Finalizar la Venta en el Botón “**Venta**”

Excepciones

Las Excepciones son creadas y Controladas a lo largo del Programa

```
1 referencia
private void bt_login_Click(object sender, EventArgs e)
{
    try
    {
        DataBase.validaUsuario(int.Parse(txbx_Usuario.Text), txbx_password.Text);
        FrmMenu menu = new FrmMenu();
        timer1.Enabled = false;
        this.Hide();
        menu.ShowDialog();
        this.Show();
        timer1.Enabled = true;
        ResetTextBoxes();
    }
    catch (Exception)
    {
        MessageBox.Show("Usuario o Clave inválidos");
        txbx_Usuario.Text = null;
        txbx_password.Text = null;
    }
}
```

Test Unitarios

`public void ValidarUsuario()` Se Valida la Excepcion cuando la clave es Incorrecta

`public void NombreRepetido()` Se valida la Excepcion cuando se crea un Producto con nombre repetido

`public void AgregarCliente()` Se Valida que se cree un Cliente Correctamente

Tipos Genéricos

```
public class Serializacion<T> where T : class
```

Implementación de Generics para Serealizar Archivos

Interfaces

```
public interface IFormato
```

Implementación de Interfaces para dar formato a los objetos, en el momento de hacer un Log de las Ventas

Archivos y Serialización

```
public class Serializacion<T> where T : class
```

Clase Genérica para Serealizar en formato XML

SQL – Base de Datos

```
public static class DataBase
```

Se conecta a la Base de Datos, para validar usuarios, actualizar datos de Productos, Alta de Clientes

Hilos

```
public partial class FrmVenta : Form
```

En el Form de las Ventas

```
Thread hiloTiempo;
```

Variable Thread de instancia

al hacer el Load del Form

```
private void TiempoVentana()
{
    if (hiloTiempo is null)
    {
        this.hiloTiempo = new Thread(EtiquetaTiempo);
        hiloTiempo.Start();
    }
    else if (hiloTiempo.IsAlive)
    {
        hiloTiempo.Abort();
        hiloTiempo = new Thread(EtiquetaTiempo);
        hiloTiempo.Start();
    }
}
```

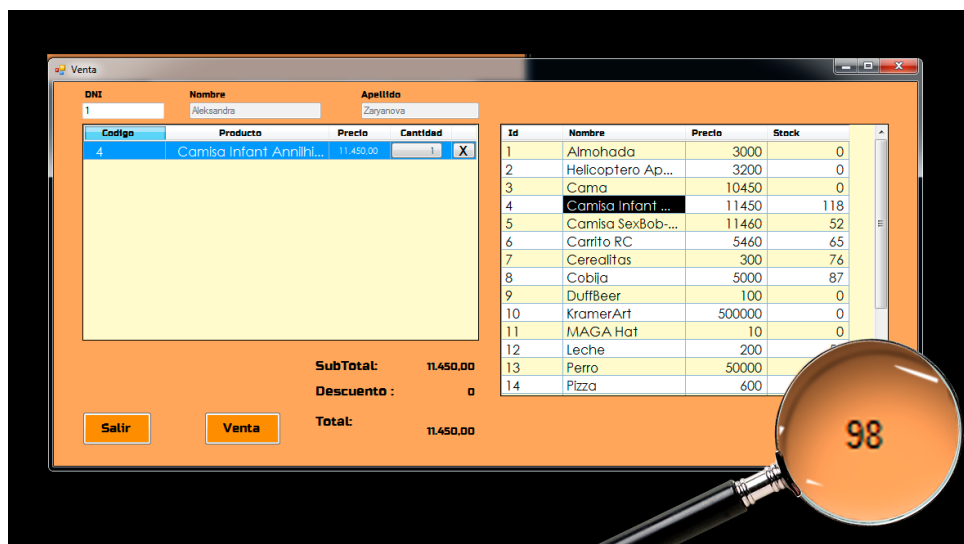
Se instancia el hilo y comienza su ejecución

```
private void EtiquetaTiempo()
{
    if (this.lbTiempo.InvokeRequired)
    {
        this.lbTiempo.BeginInvoke((MethodInvoker)delegate ()
        {
            this.lbTiempo.Text = (int.Parse(this.lbTiempo.Text) + 1).ToString();

        });
    }
    else
    {
        this.lbTiempo.Text = (int.Parse(this.lbTiempo.Text) + 1).ToString();
    }

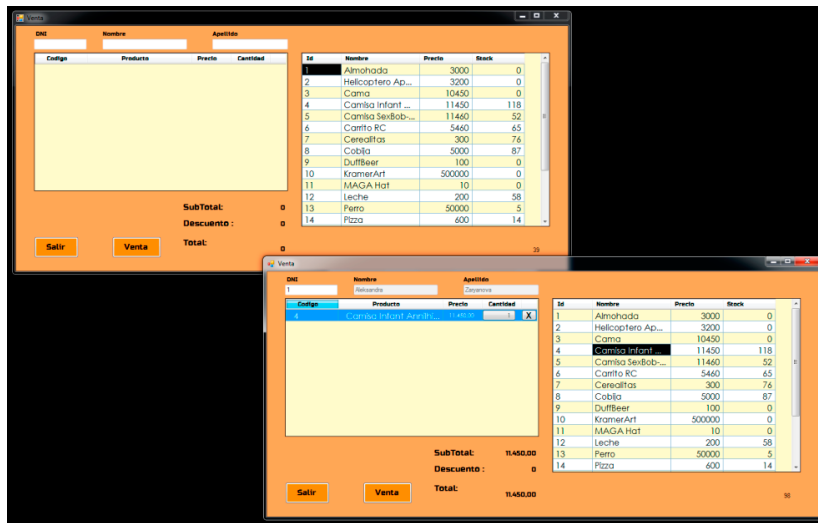
    Thread.Sleep(1000);
    EtiquetaTiempo();
}
```

Crear un loop, que actualiza el tiempo que la Ventana está abierta



Eventos

Cuando se abren 2 o más ventanas de Venta y se Finaliza una venta en alguna de las Ventanas, se ejecuta el Evento que actualiza todas las listas de las Ventanas Activas



```
public partial class FrmMenu : Form
{
    //Evento Refrescar Ventanas
    public delegate void RefrescarVentanas();
    public event RefrescarVentanas VentanasRefresh;
}
```

```
public partial class FrmVenta : Form
{
    //Evento Refrescar Grids
    public delegate void RefreshGrids();
    public event RefreshGrids ActivarRefresh;
}
```

```
private void button2_Click(object sender, EventArgs e)
{
    FrmVenta frmVenta = new FrmVenta(this);
    //Asocio el Evento en la instancia del nuevo Form
    frmVenta.ActivarRefresh += RefrescarGrids;
    frmVenta.Show();
}
```

```
public FrmVenta(FrmMenu parent)// Constructor con el parent para referencia
{
    InitializeComponent();
    //Asocio el Evento en el constructor, para mantener referencia del
    Form Parent(frmMenu)
    parent.VentanasRefresh -= VentanasGridRefresh;
    parent.VentanasRefresh += VentanasGridRefresh;
}
```

El Evento se Ejecuta cuando se presiona el botón Venta.

Métodos de Extensión

Se creo un método para adjuntar las ventas realizadas en un archivo .txt

```
public static class ExtensionArchivo
```

Se implementa en el Método de Clase de “Entidades.Establecimiento”

```
public static void HacerVenta(string idCliente, List<Producto> productos)
{
    Bridge aux;

    ventas.Add(new Venta(SetVentaNum(), Establecimiento.GetCliente(int.Parse(idCliente)),
productos));
    RemoverStock(productos);
    ventas[ventas.Count - 1].VentaLog();
    aux = new Bridge(Establecimiento.Ventas);
    Serializacion<Bridge>.Guardar(aux);
}
```