**NC(V) COMPUTER PROGRAMMING LEVEL 4 – PRACTICAL ASSESSMENT TASK (PAT) 2025**

**STUDENT NAME : Phasha Ntheno Edwin**  
**BUSINESS NAME:** Ko Stick (Tuck-shop selling grocery items)  
**ASSESSMENT YEAR:** 2025 (Tuck-shop)

**1. BUSINESS BACKGROUND**

**Business Name:** Ko Stick  
**Location:** 37648 Konyane Street, Ext 22, Pretoria, 0122  
**Nature:** Ko Stick is a small grocery tuck-shop that began operation in 2022. The business is family-owned and hand-prepares kotas using in-house bread. Ko Stick sources ingredients like potatoes from Marabastad and meat products from Roots. The tuck-shop also offers extras such as chicken legs and pork ribs. Orders are written down manually to track sales.

Ko Stick’s popular menu items include:

* **Kotas:** Beefy Bacon Kota, Rib Kota, Munchie Filler Kota
* **Sandwiches:** Bacon and Egg Toasted
* **Dagwoods:** Rib and Bacon Dag, Bacon Avo Dag
* **Extras:** Chicken Legs, Fish, Pork Ribs

**2. IDENTIFIED BUSINESS PROCESSES**

**📌 Process 1: Order Management**

* Customers place orders by selecting menu items.
* Orders are written on paper to track and serve in sequence.
* Problem: Manual tracking is prone to loss/errors, especially during busy hours.

**📌 Process 2: Stock Tracking**

* Ingredients like bread, potatoes, and meat are used daily.
* Stock levels are not automatically updated; tracking is manual.
* Problem: Items can run out mid-day without warning, affecting order fulfillment.

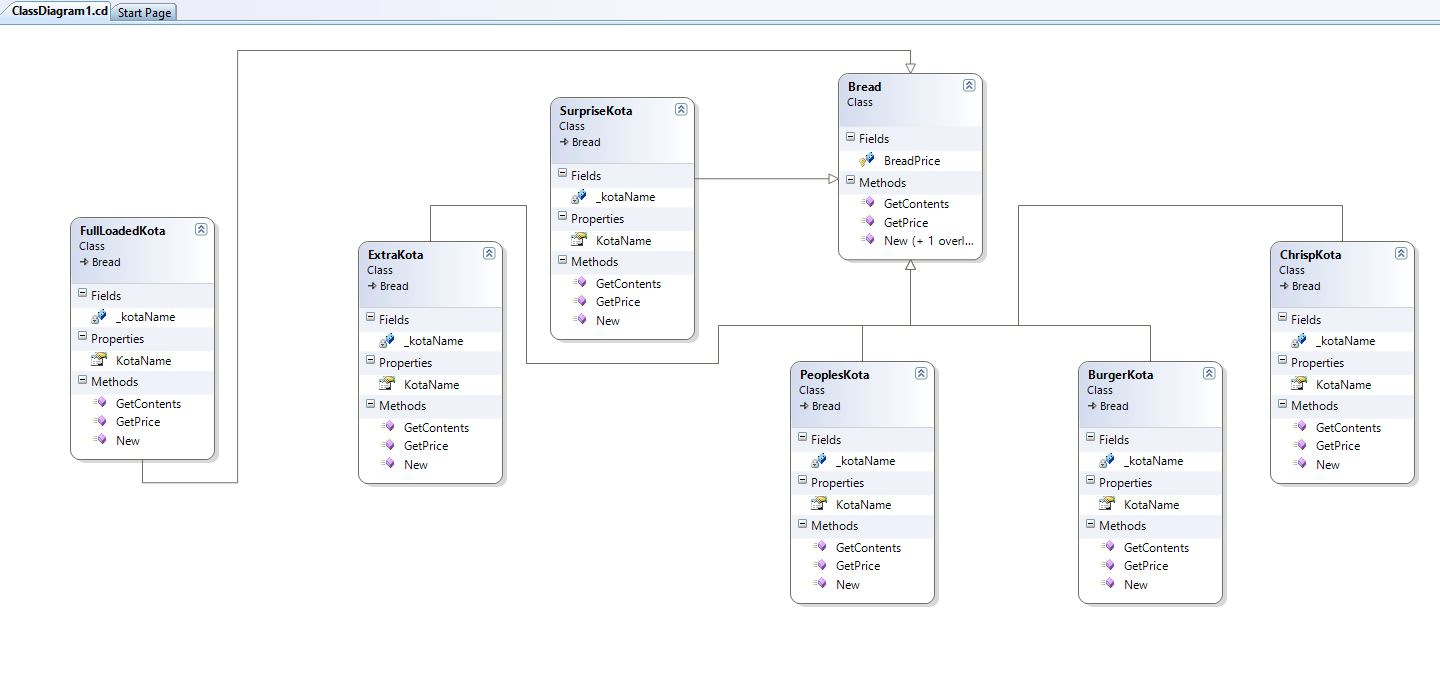
**3. SOFTWARE SOLUTION DESIGN**

To address these two critical issues, I will develop a **custom software solution** that includes:

1. **Digital Order System** with UI and item selection.
2. **Live Stock Tracking Form** linked to ingredients.

This project uses **VB.NET** and applies **Object-Oriented Programming (OOP)** principles.

**4. CLASS DIAGRAM**

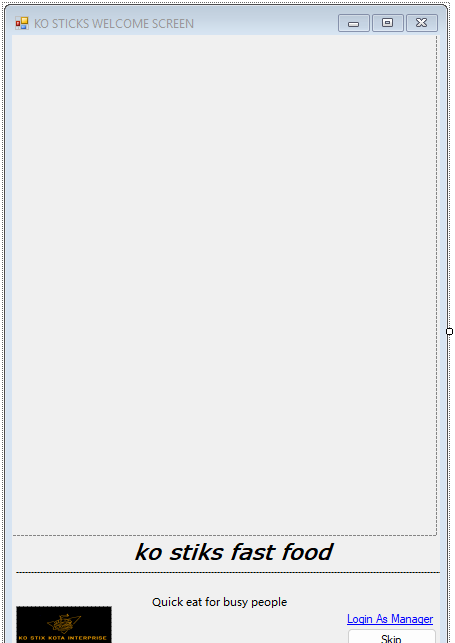
****Relationships:

* Kota inherits from Bread
* Extras are associated with Kota selections

**5. SOLUTION IMPLEMENTATION**

**✅ User Interface (Forms):**

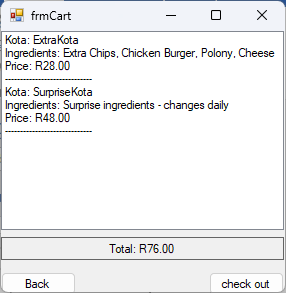
1. **Form 1:** Welcome/Login with Skip and Manager Login

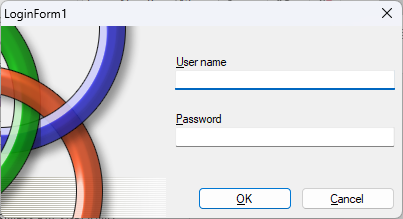
Without debugging 



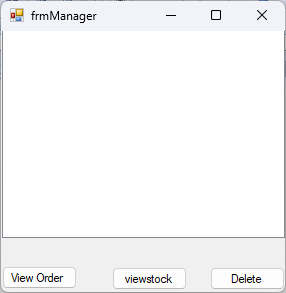
1. **Form 2:** Home Kota selection



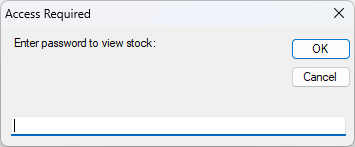
1. **Form 3:** View cart, total cost, confirm order
2. 
3. **Form 4:** Extras Form – log in as manager



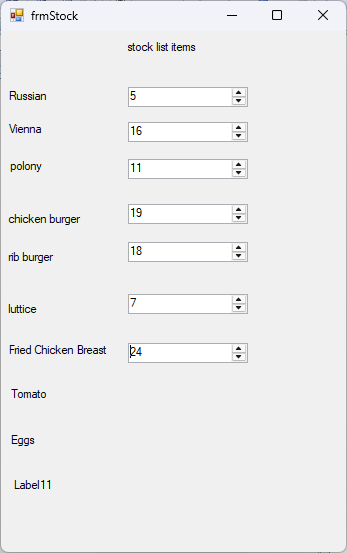
1. **Form 5:** - manager form to receive orders



1. **INPUT BOX -**REQUIRED TO VIEW STOCK



1. **Form 7:** STOCK FORM



**✅ Functionality:**

* Kotas inherit bread price (¼ of loaf)
* Extras added dynamically to total
* NumericUpDown controls manage stock items
* Confirmation prompts and error messages for invalid input

**✅ Code Requirements:**

* Constructors for all classes
* Getter/Setter properties
* Auxiliary functions
* Validation

**6. code implementation**

**Classes**

**Super class of bread**

Public Class Bread

Protected BreadPrice As Double

Dim \_ExpiryDate As Date

' Default Constructor

Public Sub New()

BreadPrice = 12

End Sub

' Parameterized Constructor

Public Sub New(ByVal price As Double)

If price <= 0 Then

Throw New ArgumentException("Bread price must be greater than zero.")

End If

BreadPrice = price

End Sub

' Method Modifiers: Overridable

Public Overridable Function GetPrice() As Double

Return BreadPrice / 4 ' Each kota uses 1/4 bread

End Function

Public Overridable Function GetContents() As String

Return "Bread"

End Function

' Auxiliary Method

Public Function IsExpired() As Boolean

Return \_ExpiryDate < Date.Today

End Function

End Class

**BurgerKota**

Public Class BurgerKota

Inherits Bread

Private \_kotaName As String = "Burger Kota"

Public Property KotaName() As String

Get

Return \_kotaName

End Get

Set(ByVal value As String)

\_kotaName = value

End Set

End Property

Public Sub New()

MyBase.New(12)

End Sub

Public Overrides Function GetPrice() As Double

Return 35 + MyBase.GetPrice() ' kota price + bread price devided by 4

End Function

Public Overrides Function GetContents() As String

Return "Double Chicken Burger, Rib Burger, Cheese, Onion, Lettuce"

End Function

End Class

**ChrispKota**

Public Class ChrispKota

Inherits Bread

Private \_kotaName As String = "Chrisp Kota"

Public Property KotaName() As String

Get

Return \_kotaName

End Get

Set(ByVal value As String)

\_kotaName = value

End Set

End Property

Public Sub New()

MyBase.New(12)

End Sub

Public Overrides Function GetPrice() As Double

Return 50 + MyBase.GetPrice() ' kota price + bread price devided by 4

End Function

Public Overrides Function GetContents() As String

Return "Fried Chicken Breast, Tomato, Chips, Onion, Lettuce, Cheese, Polony"

End Function

End Class

**BurgerKota**

Public Class BurgerKota

Inherits Bread

Private \_kotaName As String = "Burger Kota"

Public Property KotaName() As String

Get

Return \_kotaName

End Get

Set(ByVal value As String)

\_kotaName = value

End Set

End Property

Public Sub New()

MyBase.New(12)

End Sub

Public Overrides Function GetPrice() As Double

Return 35 + MyBase.GetPrice() ' kota price + bread price devided by 4

End Function

Public Overrides Function GetContents() As String

Return "Double Chicken Burger, Rib Burger, Cheese, Onion, Lettuce"

End Function

End Class

**ExtraKota**

Public Class ExtraKota

Inherits Bread

Private \_kotaName As String = "Extra Kota"

Public Property KotaName() As String

Get

Return \_kotaName

End Get

Set(ByVal value As String)

\_kotaName = value

End Set

End Property

Public Sub New()

MyBase.New(12)

End Sub

Public Overrides Function GetPrice() As Double

Return 25 + MyBase.GetPrice() ' kota price + bread price devided by 4

End Function

Public Overrides Function GetContents() As String

Return "Extra Chips, Chicken Burger, Polony, Cheese"

End Function

End Class

**PeoplesKota**

Public Class PeoplesKota

Inherits Bread

Private \_kotaName As String = "People's Kota"

Public Property KotaName() As String

Get

Return \_kotaName

End Get

Set(ByVal value As String)

\_kotaName = value

End Set

End Property

Public Sub New()

MyBase.New(12)

End Sub

Public Overrides Function GetPrice() As Double

Return 30 + MyBase.GetPrice() ' kota price + the bread price devided by 4

End Function

Public Overrides Function GetContents() As String

Return "Russian, Cheese, Polony, Vienna, Egg, Chips, Lettuce"

End Function

End Class

**SurpriseKota**

Public Class SurpriseKota

Inherits Bread

Private \_kotaName As String = "Surprise Kota"

Public Property KotaName() As String

Get

Return \_kotaName

End Get

Set(ByVal value As String)

\_kotaName = value

End Set

End Property

Public Sub New()

MyBase.New(12)

End Sub

Public Overrides Function GetPrice() As Double

Return 45 + MyBase.GetPrice()

End Function

Public Overrides Function GetContents() As String

Return "Surprise ingredients - changes daily"

End Function

End Class

**Code implementation of form**

**Form 1 welcome form**

Public Class Form1

Dim imagePaths() As String ' Array to hold file paths

Dim currentImageIndex As Integer = 0

Dim imageFolder As String = " G:\computer programming latest solution\ko sticks fast food\ko sticks fast food\bin\Debug/level 4"

Private Sub Form1\_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

' Get all image files in the folder

If IO.Directory.Exists(imageFolder) Then

imagePaths = IO.Directory.GetFiles(imageFolder, "\*.jpeg")

If imagePaths.Length > 0 Then

pbSlideshow.SizeMode = PictureBoxSizeMode.StretchImage

pbSlideshow.Image = Image.FromFile(imagePaths(currentImageIndex))

tmrSlideshow.Interval = 3000 ' 3 seconds

tmrSlideshow.Start()

Else

MessageBox.Show("No images found in folder.")

End If

Else

MessageBox.Show("Image folder not found: " & imageFolder)

End If

End Sub

Private Sub tmrSlideshow\_Tick(ByVal sender As Object, ByVal e As EventArgs) Handles tmrSlideshow.Tick

If imagePaths IsNot Nothing AndAlso imagePaths.Length > 0 Then

currentImageIndex += 1

If currentImageIndex >= imagePaths.Length Then

currentImageIndex = 0

End If

pbSlideshow.Image = Image.FromFile(imagePaths(currentImageIndex))

End If

End Sub

Private Sub btnskiplogin\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnskiplogin.Click

Me.Hide()

Form2.Show()

End Sub

Private Sub LinkLabel1\_LinkClicked(ByVal sender As System.Object, ByVal e As System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles LinkLabel1.LinkClicked

Me.Hide()

LoginForm1.Show()

End Sub

End Class

**Form 2 - KO STICKS ODERING LOGIC SYSTEM**

Public Class Form2

Public Shared Cart As New List(Of Bread)

Private Sub AddToCart(ByVal kota As Bread)

Dim confirm As String

confirm = MessageBox.Show("Do you want to add {kota.KotaName} to your cart?", "Confirm Add", MessageBoxButtons.YesNo)

If confirm = DialogResult.Yes Then

Cart.Add(kota)

End If

End Sub

Private Sub picFullLoadedKota\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles picFullLoadedKota.Click

AddToCart(New FullLoadedKota())

End Sub

Private Sub picExtraKota\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles picExtraKota.Click

AddToCart(New ExtraKota())

End Sub

Private Sub picPeoplesKota\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles picPeoplesKota.Click

AddToCart(New PeoplesKota())

End Sub

Private Sub picBurgerKota\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles picBurgerKota.Click

AddToCart(New BurgerKota())

End Sub

Private Sub picChrispKota\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles picChrispKota.Click

AddToCart(New ChrispKota())

End Sub

Private Sub picSurpriseKota\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles picSurpriseKota.Click

AddToCart(New SurpriseKota())

End Sub

Private Sub btnViewCart\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnViewCart.Click

Dim cartForm As New frmCart()

cartForm.CartItems = Cart

cartForm.Show()

Me.Hide()

End Sub

Private Sub btnbacktowelcomeform\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnbacktowelcomeform.Click

Me.Hide()

Form1.Show()

End Sub

End Class

**Form cart**

Imports System.Net.Sockets

Imports System.Text

Public Class frmCart

Private \_cartItems As List(Of Bread)

Public Property CartItems() As List(Of Bread)

Get

Return \_cartItems

End Get

Set(ByVal value As List(Of Bread))

\_cartItems = value

End Set

End Property

Private Sub frmCart\_Load(ByVal sender As Object, ByVal e As EventArgs) Handles MyBase.Load

lstCart.Items.Clear()

Dim total As Double = 0

For Each item In CartItems

lstCart.Items.Add("Kota: " & item.GetType().Name)

lstCart.Items.Add("Ingredients: " & item.GetContents())

lstCart.Items.Add("Price: R" & item.GetPrice().ToString("F2"))

lstCart.Items.Add("-----------------------------")

total += item.GetPrice()

Next

lblTotal.Text = "Total: R" & total.ToString("F2")

End Sub

Private Sub btnCheckout\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnCheckout.Click

' check if there is missing ingridients

Dim missingIngredients As New List(Of String)

For Each item As Bread In CartItems

Dim ingredients = item.GetContents().ToLower()

If ingredients.Contains("chips") And frmStock.nudChips.Value = 0 Then missingIngredients.Add("Chips")

If ingredients.Contains("russian") And frmStock.nudRussian.Value = 0 Then missingIngredients.Add("Russian")

If ingredients.Contains("polony") And frmStock.nudPolony.Value = 0 Then missingIngredients.Add("Polony")

If ingredients.Contains("vienna") And frmStock.nudVienna.Value = 0 Then missingIngredients.Add("Vienna")

If ingredients.Contains("cheese") And frmStock.nudCheese.Value = 0 Then missingIngredients.Add("Cheese")

If ingredients.Contains("egg") And frmStock.nudEgg.Value = 0 Then missingIngredients.Add("Egg")

If ingredients.Contains("rib burger") And frmStock.nudRibBurger.Value = 0 Then missingIngredients.Add("Rib Burger")

If ingredients.Contains("chicken burger") And frmStock.nudChickenBurger.Value = 0 Then missingIngredients.Add("Chicken Burger")

If ingredients.Contains("fried chicken") And frmStock.nudFriedChicken.Value = 0 Then missingIngredients.Add("Fried Chicken")

If ingredients.Contains("lettuce") And frmStock.nudLettuce.Value = 0 Then missingIngredients.Add("Lettuce")

If ingredients.Contains("tomato") And frmStock.nudTomato.Value = 0 Then missingIngredients.Add("Tomato")

If ingredients.Contains("onion") And frmStock.nudOnion.Value = 0 Then missingIngredients.Add("Onion")

Next

If missingIngredients.Count > 0 Then

Dim message1 As String = "These ingredients are out of stock: " & vbCrLf & String.Join(vbCrLf, missingIngredients.Distinct()) & vbCrLf & "Would you like to continue? Staff will offer a discount."

Dim result = MessageBox.Show(message1, "Stock Alert", MessageBoxButtons.YesNo)

If result = DialogResult.No Then

Exit Sub

End If

End If

' Subtract ingredients from stock

For Each item As Bread In CartItems

Dim contents = item.GetContents().ToLower()

If contents.Contains("chips") Then frmStock.nudChips.Value -= 1

If contents.Contains("russian") Then frmStock.nudRussian.Value -= 1

If contents.Contains("polony") Then frmStock.nudPolony.Value -= 1

If contents.Contains("vienna") Then frmStock.nudVienna.Value -= 1

If contents.Contains("cheese") Then frmStock.nudCheese.Value -= 1

If contents.Contains("egg") Then frmStock.nudEgg.Value -= 1

If contents.Contains("rib burger") Then frmStock.nudRibBurger.Value -= 1

If contents.Contains("chicken burger") Then frmStock.nudChickenBurger.Value -= 1

If contents.Contains("fried chicken") Then frmStock.nudFriedChicken.Value -= 1

If contents.Contains("lettuce") Then frmStock.nudLettuce.Value -= 1

If contents.Contains("tomato") Then frmStock.nudTomato.Value -= 1

If contents.Contains("onion") Then frmStock.nudOnion.Value -= 1

Next

Dim message As New StringBuilder()

For Each item As Bread In CartItems

message.AppendLine("Kota: " & item.GetType().Name)

message.AppendLine("Ingredients: " & item.GetContents())

message.AppendLine("Price: R" & item.GetPrice().ToString("F2"))

message.AppendLine("-----------------------------")

Next

' Send to manager

Try

Dim client As New Net.Sockets.TcpClient("192.168.1.100", 5000) ' <-- manager PC IP

Dim stream As Net.Sockets.NetworkStream = client.GetStream()

Dim data() As Byte = Encoding.ASCII.GetBytes(message.ToString())

stream.Write(data, 0, data.Length)

stream.Close()

client.Close()

Catch ex As Exception

MessageBox.Show("Failed to send order: " & ex.Message)

End Try

' Clear cart

CartItems.Clear()

lstCart.Items.Clear()

lblTotal.Text = "Total: R0.00"

MessageBox.Show("Thank you for your order!")

End Sub

Private Sub btnBacktomain\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnBacktomain.Click

Me.Hide()

Form2.Show()

End Sub

End Class

**Login form**

Public Class LoginForm1

Private Sub Cancel\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btncalcel.Click

Me.Close()

End Sub

Private Sub btnok\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnok.Click

Dim password As String = "1234"

Dim ussrname As String = "Phasha"

If txtpassword.Text = password And txtuusername.Text = ussrname Then

Me.Hide()

frmManager.Show()

Else

MessageBox.Show(" incorrect password ")

End If

End Sub

End Class

**Manager form**

Imports System.Net

Imports System.Net.Sockets

Imports System.Text

Imports System.Threading

Public Class frmManager

Private managerListener As TcpListener

Private serverThread As Thread

Private Sub btnViewOder\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnViewOder.Click

Try

Dim listener As New TcpListener(IPAddress.Any, 5000)

listener.Start()

MessageBox.Show("Waiting for order...")

Dim client As TcpClient = listener.AcceptTcpClient()

Dim stream As NetworkStream = client.GetStream()

Dim buffer(4096) As Byte

Dim bytesRead As Integer = stream.Read(buffer, 0, buffer.Length)

Dim message As String = Encoding.ASCII.GetString(buffer, 0, bytesRead)

lstOrders.Items.Clear()

lstOrders.Items.Add("=== New Order ===")

' Directly split and add to listbox

For Each item As String In message.Split(CChar(CStr(StringSplitOptions.RemoveEmptyEntries)))

lstOrders.Items.Add(item)

Next

lstOrders.Items.Add("-----------------------------")

stream.Close()

client.Close()

listener.Stop()

Catch ex As Exception

MessageBox.Show("Error receiving order: " & ex.Message)

End Try

End Sub

Private Sub btndelete\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btndelete.Click

If lstOrders.SelectedIndex <> -1 Then

Dim confirm = MessageBox.Show("Do you want to delete this item?", "Delete Item", MessageBoxButtons.YesNo)

If confirm = DialogResult.Yes Then

lstOrders.Items.RemoveAt(lstOrders.SelectedIndex)

End If

Else

MessageBox.Show("Select an item to delete.")

End If

End Sub

Private Sub btnstock\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnstock.Click

Dim password As String = InputBox("Enter password to view stock:", "Access Required")

If password = "1234" Then

Me.Hide()

frmStock.Show()

Else

MessageBox.Show("Incorrect password!", "Access Denied")

End If

End Sub

End Class

**Stock form**

Public Shared nudChips As New NumericUpDown()

Public Shared nudRussian As New NumericUpDown()

Public Shared nudPolony As New NumericUpDown()

Public Shared nudVienna As New NumericUpDown()

Public Shared nudCheese As New NumericUpDown()

Public Shared nudEgg As New NumericUpDown()

Public Shared nudRibBurger As New NumericUpDown()

Public Shared nudChickenBurger As New NumericUpDown()

Public Shared nudFriedChicken As New NumericUpDown()

Public Shared nudLettuce As New NumericUpDown()

Public Shared nudTomato As New NumericUpDown()

Public Shared nudOnion As New NumericUpDown()