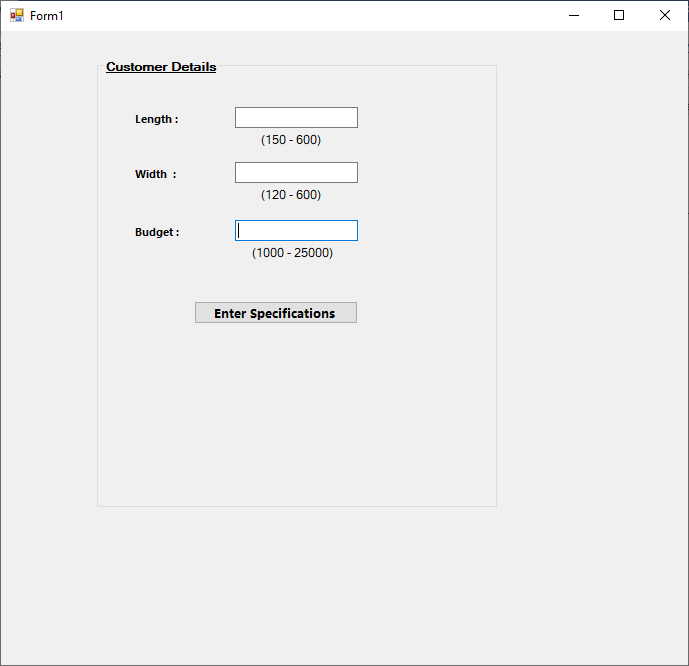
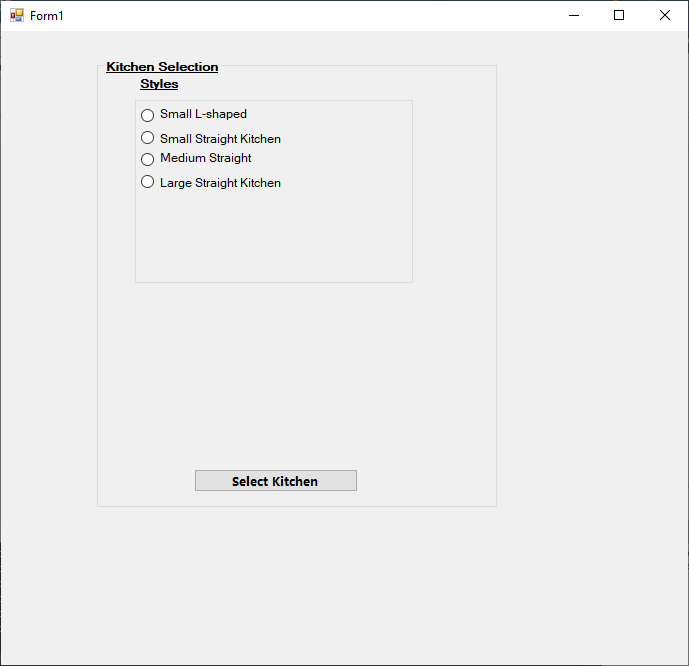
**Application Programming Project**

# Screenshots of the user Interface

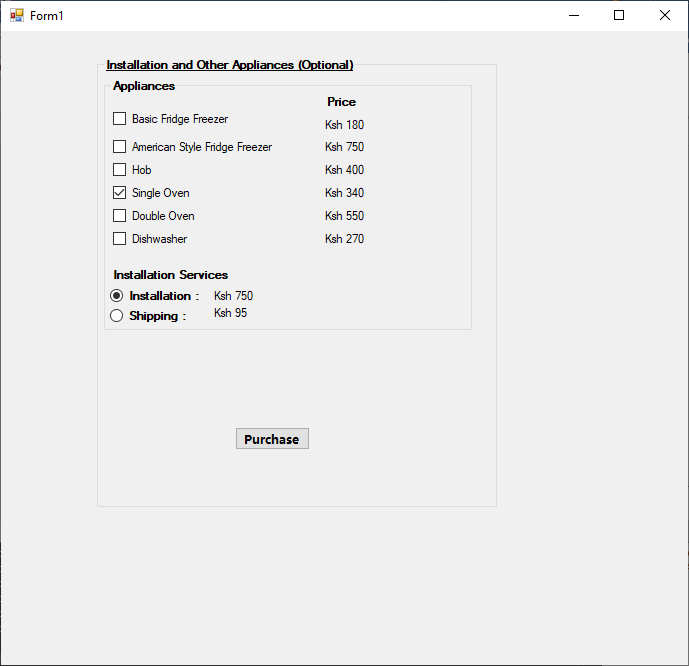
Customer details



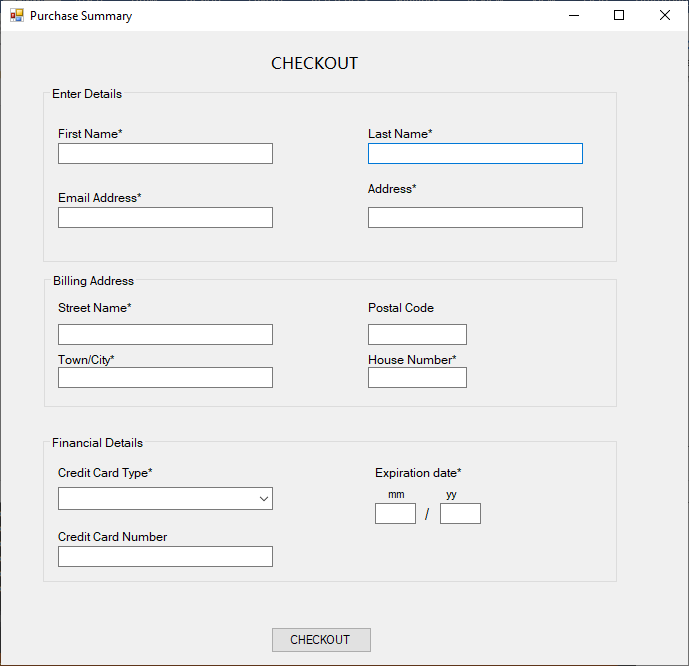
Kitchen Selection



Appliances selection and installation services



Checkout



# Pseudocode

# Completed Object Definition Sheet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Type | Property | Value | Event Procedure |
| 1 | Form | Name | Form1  Form2 | Load()  btnProceed1.Click() |
|  |  | Text | Form1  Form2 |  |
|  |  |  |  |  |
| 2 | Group Box | Name | gbxCustomerDetails  gbxKitchenStyles  gbxAppliancesOptions  gbxAmount  gbxAppliances  gbxCountertops  gbxBox |  |
|  |  | Text | Customer Details  Kitchen Selection  Installation and Other Appliances (Optional)  Amount |  |
|  |  |  |  |  |
| 3 | Labels | Name | lblRange1  lblRange2  lblRange3  lblWidth  lblLength  ldlBudget  lblRemarks  lblShipping  lblPrice  lblTotalCost  lblTotal  lblInstallation  lblInstallationPrice  lblRemind  lblStyles  lblKsh1  lblKsh2  lblKsh3  lblKsh4  lblKsh5  lblKsh6  lblKsh7  lblKsh8  lblKsh9  lblKsh10  lblGranitePrice  lblWoodPrice  lblLaminatePrice  lblPrices |  |
| 4 | Buttons | Name | btnEnter  btnSelect  btnProceed1  btnPurchase |  |
|  |  | Text | Enter Specifications  Select kitchen  Proceed  Purchase |  |
| 5 | Text Box | Name | txbrmBudget  txbrmWidth  txbrmLength |  |
| 6 | Radio Box | Name | rdoShipping  rdoInstallation  rdbGranite  rdbWoodBlk  rdbLaminate |  |
| 7 | Check Box | Name | cboApp1  cboApp2  cboApp3  cboApp4  cboApp5  cboApp6 |  |
| 8 |  |  |  |  |

# The Visual Basic program code

Form 1

Public Class Form1

'creating all arrays and variables for holding data

Dim kitchenStyle() As String = {"Small L-shaped Kitchen", "Medium L-shaped Kitchen", "Large L-shaped Kitchen", "Small Straight Kitchen", "Medium Straight Kitchen", "Large Straight Kitchen", "Large Island Kitchen"}

Dim prices() As Integer = {3500, 4500, 5750, 6500, 7750, 9500, 8250, 9500, 12500, 1000, 1500, 2200, 5250, 6200, 8000, 6750, 8500, 11000, 12500, 15000, 23000}

Dim appliances() As String = {"Basic Fridge Freezer", "American Style Fridge Freezer", "Hob", "Single Oven", "Double Oven", "Dishwasher"}

Dim appliancePrices() As Integer = {180, 750, 400, 340, 550, 270}

Dim installationCosts() As Integer = {1250, 2750, 3750, 750, 1500, 2500, 4750}

Dim installationPrice As Integer

Dim Counterpricelist(3) As Integer

Dim chosenStyle As String

Dim total, totalCost As Double

'creating a function to validate date and ensure it is within bounds given

Function validateData(ByVal lnth As Integer, ByVal width As Integer, ByVal budget As Integer) As Boolean

'This function validates data with an if statement and returns true for incorrect input and false for correct input

If (lnth < 150 Or width < 20 Or budget < 1000) Or (lnth > 600 Or width > 600 Or budget > 25000) Then

Return True

Else

Return False

End If

End Function

Function verify() As String

'this function verifies input by ensuring no characters except numbers can be accepted

'create a msg variable to hold return type message

Dim length, width, budget As Integer

Dim msg As String

'initialize with an empty text as default

msg = " "

'catch and raise exceptions with a try statement

Try

length = rmLength.Text

width = rmWidth.Text

budget = rmBudget.Text

'throw an exception when validateData() function returns as True

If (validateData(length, width, budget)) Then

Throw New Exception("Enter values within applicable ranges")

End If

Catch ex As InvalidCastException

'this code catches all conversion error exceptions that could break the code

msg = "Enter all values correctly"

Return msg

Catch ez As Exception

'this one catches all other exceptions including our own

msg = ez.Message

Return msg

End Try

'return the final message

'an empty string means a correct validation has occured

Return msg

End Function

'creating a function that gets kitchen styles relating to dimensions given and budget

Function getKitchenByDIm(ByVal length As Integer, ByVal width As Integer, ByVal budget As Integer, ByVal prices() As Integer, ByVal kitchenstyle() As String,

ByVal lengthArr() As Integer, ByVal widthArr() As Integer) As String()

'create an array for storing the styles within customer's range

Dim chosenStyles(7) As String

'create a variable for holding the least amount payable for a kitchen style

Dim lowest() As Integer = {prices(0), prices(3), prices(6), prices(9), prices(12), prices(15), prices(18)}

Dim cnt As Integer = 0

Dim cntprice As Integer = 0

For counter = 0 To 6

If (length > lengthArr(counter) And width > widthArr(counter)) And budget > lowest(counter) Then

chosenStyles(cnt) = kitchenstyle(counter)

cnt += 1

End If

Next

Return chosenStyles

End Function

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles btnEnter.Click

Dim lengthsArray() As Integer = {250, 350, 450, 150, 350, 500, 450}

Dim widthsArray() As Integer = {180, 240, 350, 60, 60, 60, 450}

Dim msg As String

msg = verify()

lblRemarks.Text = msg

'checking if empty statement came back

If (msg <> (" ")) Then

'if not empty the form stays as is

gbxCustomerDetails.Visible = True

Else

'else new groupbox is loaded and made visible

gbxKitchenStyles.Visible = True

'get all kitchen styles applicable to data given

Dim chosen() As String = getKitchenByDIm(rmLength.Text, rmWidth.Text, rmBudget.Text, prices, kitchenStyle, lengthsArray, widthsArray)

Dim top, left As Integer

top = 10

left = 6

For counter = 0 To (chosen.Length - 1)

'create radiobuttons dynamically for user

Dim rdb As New RadioButton()

rdb.Text = chosen(counter)

rdb.Top = top

rdb.Left = left

rdb.Width = 147

rdb.Height = 25

gbxBox.Controls.Add(rdb)

top += 22

If (rdb.Text.Equals("")) Then

gbxBox.Controls.Remove(rdb)

End If

Next

End If

End Sub

'creating a function to get Kitchen countertop prices using the name of the chosen style

Function getKitchenPrices(ByVal prices() As Integer, ByVal text As String, ByVal styles() As String) As Integer()

Dim n, pricelist(3) As Integer

n = 0

chosenStyle = text

If (text.Equals(styles(0))) Then

For counter = 0 To 2

pricelist(n) = prices(counter)

n += 1

Next

ElseIf (text.Equals(styles(1))) Then

For counter = 3 To 5

pricelist(n) = prices(counter)

n += 1

Next

ElseIf (text.Equals(styles(2))) Then

For counter = 6 To 8

pricelist(n) = prices(counter)

n += 1

Next

ElseIf (text.Equals(styles(3))) Then

For counter = 9 To 11

pricelist(n) = prices(counter)

n += 1

Next

ElseIf (text.Equals(styles(4))) Then

For counter = 12 To 14

pricelist(n) = prices(counter)

n += 1

Next

ElseIf (text.Equals(styles(5))) Then

For counter = 15 To 17

pricelist(n) = prices(counter)

n += 1

Next

ElseIf (text.Equals(styles(6))) Then

For counter = 18 To 20

pricelist(n) = prices(counter)

n += 1

Next

End If

Return pricelist

End Function

'creating a sub to load countertops as radio buttons when user moves cursor over the groupbox containing styles after selection

Private Sub loadCounters(sender As Object, e As EventArgs) Handles gbxBox.MouseMove

Dim rdb As New RadioButton()

'dynamic creation of radio buttons at runtime

For Each rdb In gbxBox.Controls

If rdb.Checked Then

gbxCountertops.Visible = True

'add prices for countertops using getKitchenPrices () function

Dim priceofcountertops() As Integer = getKitchenPrices(prices, rdb.Text, kitchenStyle)

Counterpricelist = priceofcountertops

lblLaminatePrice.Text = priceofcountertops(0)

lblWoodPrice.Text = priceofcountertops(1)

lblGranitePrice.Text = priceofcountertops(2)

End If

Next

End Sub

'creating a function to get the installation costs of the kitchen Style

Function getInstallationCost() As Integer

For counter = 0 To 6

If chosenStyle.Equals(kitchenStyle(counter)) Then

installationPrice = installationCosts(counter)

End If

Next

Return installationPrice

End Function

Function getItemCost(ByVal itemname As String) As Integer

Dim itemCost As Integer

For counter = 0 To 5

If itemname.Equals(appliances(counter)) Then

itemCost = appliancePrices(counter)

End If

Next

Return itemCost

End Function

'''when select kitchen button is pressed the following sub handles the event

'''the program checks if any of the given countertops is selected

'''and records the price of the selected item

'''the routine also handles no input with an error message

Private Sub btnProceed\_Click(sender As Object, e As EventArgs) Handles btnSelect.Click

Dim counterTopPrice As Integer = 0

lblRemind.Text = ""

If gbxCountertops.Visible Then

total = 0

If rdbLaminate.Checked Then

counterTopPrice = Counterpricelist(0)

gbxAppliancesOptions.Visible = True

total = counterTopPrice

lblTotalCost.Text = $"{total}"

lblInstallationPrice.Text += $"{getInstallationCost()}"

ElseIf rdbWoodBlk.Checked Then

counterTopPrice = Counterpricelist(1)

gbxAppliancesOptions.Visible = True

total = counterTopPrice

lblTotalCost.Text = $"{total}"

lblInstallationPrice.Text += $"{getInstallationCost()}"

ElseIf rdbGranite.Checked Then

counterTopPrice = Counterpricelist(2)

gbxAppliancesOptions.Visible = True

total = counterTopPrice

lblTotalCost.Text = $"{total}"

lblInstallationPrice.Text += $"{getInstallationCost()}"

Else

lblRemind.Text = "Please select a countertop to proceed"

End If

Else

lblRemind.Text = "Please select a kitchen style to proceed"

'add more code

End If

End Sub

''' <summary>

''' this sub refreshes the price whenever the Purchase button is pressed

''' it adds the prices of all selected appliances

''' it also ensures the user selects an option between shipping and installation

''' it handles this with an error message

''' </summary>

Private Sub refreshPrice() Handles btnPurchase.Click

Dim cost As Integer = 0

lblShipping.Text = ""

If cboApp1.Checked Then

cost += getItemCost(cboApp1.Text)

End If

If cboApp2.Checked Then

cost += getItemCost(cboApp2.Text)

End If

If cboApp3.Checked Then

cost += getItemCost(cboApp3.Text)

End If

If cboApp4.Checked Then

cost += getItemCost(cboApp4.Text)

End If

If cboApp5.Checked Then

cost += getItemCost(cboApp5.Text)

End If

If cboApp6.Checked Then

cost += getItemCost(cboApp6.Text)

End If

If rdoInstallation.Checked Then

cost += getInstallationCost()

gbxAmount.Visible = True

btnProceed1.Visible = True

ElseIf rdoShipping.Checked Then

cost += 95

gbxAmount.Visible = True

btnProceed1.Visible = True

Else

lblShipping.Text = "Please pick between Shipping and Installation"

End If

totalCost = total + cost

lblTotalCost.Text = $"{totalCost}"

End Sub

''' <summary>

''' te last sub that runs when proceed button is clicked

''' it hides the current form and takes the user to another form

''' </summary>

Private Sub btnProceed1\_Click(sender As Object, e As EventArgs) Handles btnProceed1.Click

Me.Hide()

Form2.Show()

End Sub

End Class

Form 2

Public Class Form2

Dim msg As String = "Enter all details Correctly"

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

If checkEmptyFields() Then

Dim amount As Double = Form1.lblTotalCost.Text

Dim output As String

output = $" CUSTOM KITCHENS {vbCrLf}{vbCrLf}Name : {txbFName.Text} {txbLName.Text} {vbCrLf}Amount : {amount}{vbCrLf}"

Dim filePath As String

filePath = System.IO.Path.Combine(

My.Computer.FileSystem.SpecialDirectories.Desktop, "KitchenReceipt.txt")

My.Computer.FileSystem.WriteAllText(filePath, output, False)

MsgBox("Purchase Completed")

Me.Close()

Form1.Close()

Else

lblError.Text = msg

End If

End Sub

''' <summary>

''' creating a function that verifies all inputs have no blanks

''' and verifies and validates all numeric inputs

''' </summary>

''' <returns></returns>

Function checkEmptyFields() As Boolean

Dim fname, lname, email, address, town, street, house, cardType As String

Dim month, year, postal As Integer

Dim cardNumber As Long

Try

fname = txbFName.Text

lname = txbLName.Text

email = tbxEmail.Text

address = tbxaddress.Text

town = tbxTown.Text

street = tbxStreet.Text

house = tbxHouseNumber.Text

cardType = cboCreditCard.Text

month = tbxMonth.Text

year = tbxYear.Text

postal = tbxPostal.Text

cardNumber = txbCreditNumber.Text

If (fname.Equals("") Or lname.Equals("") Or email.Equals("") Or address.Equals("") Or town.Equals("") Or street.Equals("") Or house.Equals("")) Then

Throw New Exception("No blank fileds")

End If

If month > 12 Or year < 20 Then

Throw New Exception("Enter correct date")

End If

If cardNumber < 1000000000000000 Or cardType.Equals("") Then

Throw New Exception("Enter correct card details")

End If

Catch ex As InvalidCastException

msg = "Please fill in all details correctly"

Return False

Catch ex As Exception

msg = ex.Message

Return False

End Try

Return True

End Function

End Class

# Simple test criteria

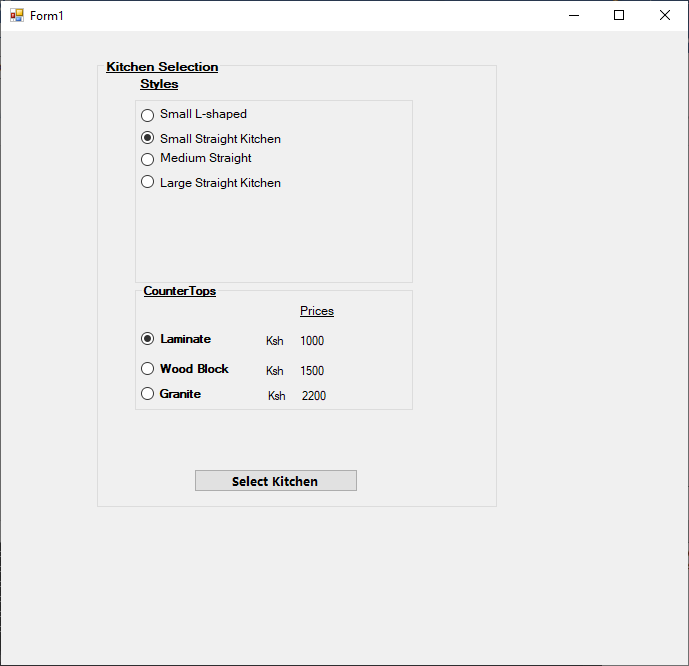
# Completed Test Log

# Screenshots of the output generated when the program code is run

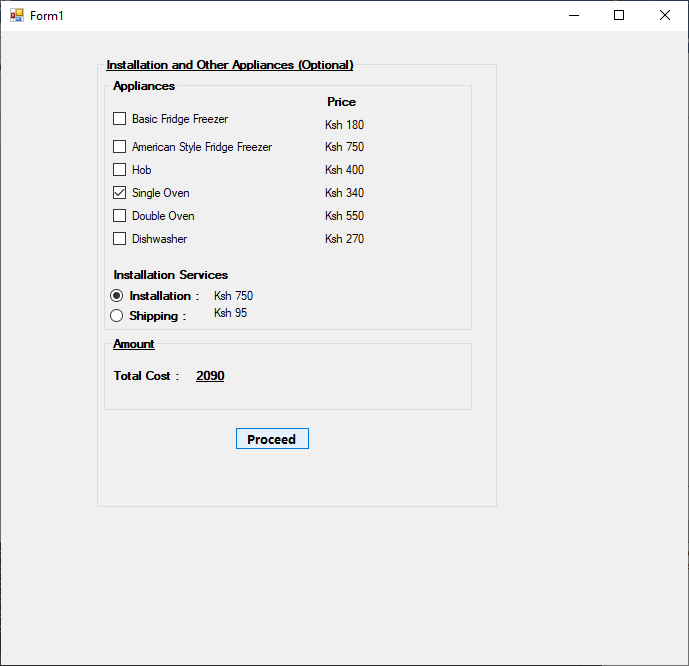
Customer Information

# 

Kitchen Style Selection



Installation and appliances



Checkout page

