

**MS806 – Business Application Programming**

**Student Name: Shreyas Lokanath**

**NUI Galway ID Number: 17231972**

**Assignment:** **5**

**Date of Submission: 22/11/2017**

MS806 – Business Application Programming

Student Name**:** Shreyas Lokanath

NUI Galway ID Number**:** 17231972

Assignment**:** 5

Date of Submission**:** 22**/**11**/**2017

/\* Student Name: Shreyas Lokanath

\* Student ID: 17231972

\* Date:9/11/2017

\* Assignment: 5

\* Task: create an application for Mr.Bagel application

\* to book bagel types and bagel sizes along with summary and

\* management report

\*/

//using namespaces and classes

**using** System**;**

**using** System**.**Collections**.**Generic**;**

**using** System**.**ComponentModel**;**

**using** System**.**Data**;**

**using** System**.**Drawing**;**

**using** System**.**Linq**;**

**using** System**.**Text**;**

**using** System**.**Threading**.**Tasks**;**

**using** System**.**Windows**.**Forms**;**

**using** System**.**Text**.**RegularExpressions**;**

**using** System**.**IO**;**

**using** System**.**Collections**;**

**using** System**.**Globalization**;**

**using** System**.**Diagnostics**;**

**namespace** Assignment5

**{**

**public** **partial** class Form1 **:** Form

**{**

**public** Form1**()**

**{**

InitializeComponent**();**

**}**

//declaration of global variables

int bagelTypeIndex**;**

int bagelSizeIndex**;**

decimal bagelCost**;**

int bagelQuantity**;**

string enterBagelQuantity**;**

int quantityOfBagel**;**

int transactionNumber **=** 0**;**

int totalTransactionCount**;**

int totalBagelsSold**;**

decimal totalValue**;**

decimal averageTransactions**;**

//declaring number of rows and columns for array storage

const int rowOfBagels **=** 16**,** columnOfBagels **=** 5**;**

//declaring file creation/existing locations

const string bagelStockFileLocation **=** @"F:/MS806-BusinessApplicationProgramming/Assignments/Assignment5-22-11-2017/Assignment5/Assignment5/bin/Debug/bagelStockFile.txt"**;**

const string bagelPriceFileLocation **=** @"F:/MS806-BusinessApplicationProgramming/Assignments/Assignment5-22-11-2017/Assignment5/Assignment5/bin/Debug/bagelPriceFile.txt"**;**

const string managementReportFileLocation **=** @"F:/MS806-BusinessApplicationProgramming/Assignments/Assignment5-22-11-2017/Assignment5/Assignment5/bin/Debug/managementReportFile.txt"**;**

//array of strings of bagel types

**readonly** static string**[]** bagelTypes **={**"Halloumi"**,**"Bangkok"**,**"ChickenPhilly"**,**"Classic Club"**,**

"Kiltimagh Special"**,**"Veggie"**,**"Ploughmans"**,**"Mexicana"**,**"Triple Cheese"**,**

"Atlantic Way"**,**"Breakfast"**,**"Maui"**,**"Classic"**,**"Chicken Caesar"**,**

"Student Surprise"**,**"Cajun" **};**

//array of strings holding bagel sizes

**readonly** static string**[]** bagelSizes **=** **{** "small"**,** "Medium"**,** "Regular"**,** "Large"**,** "Extra Large" **};**

//array of bagel stocks

int**[,]** bagelStock **=** **new** int**[**rowOfBagels**,** columnOfBagels**];**

//array of temporary bagel stocks

int**[,]** tempBagelStock **=** **new** int**[**rowOfBagels**,** columnOfBagels**];**

//array of bagel prices

decimal**[,]** bagelPrice **=** **new** decimal**[**rowOfBagels**,** columnOfBagels**];**

//method to check the bagel chosen to populate the text box with its price

**private** void ChosenBagelPrice**(object** sender**,** EventArgs e**)**

**{**

//checks the selected index of both list box

**if** **((**bagelTypeListBox**.**SelectedIndex **!=** **-**1**)** **&&** **(**bagelSizeListBox**.**SelectedIndex **!=** **-**1**))**

**{**

//number of the selected index of listbox

bagelTypeIndex **=** bagelTypeListBox**.**SelectedIndex**;**

bagelSizeIndex **=** bagelSizeListBox**.**SelectedIndex**;**

//text of the selected index

string bagelType **=** bagelTypes**[**bagelTypeIndex**];**

string bagelSize **=** bagelSizes**[**bagelSizeIndex**];**

//display the bagel type, bagel size and cost in the text box with currency conversion

bagelCost **=** bagelPrice**[**bagelTypeIndex**,** bagelSizeIndex**];**

bagelPriceTextBox**.**Text **=** bagelCost**.**ToString**(**"c"**);**

bagelTypeTextBox**.**Text **=** bagelType**;**

bagelSizeTextBox**.**Text **=** bagelSize**;**

//disable and focus the following controls

bagelQuantityTextBox**.**Enabled **=** **true;**

bagelQuantityTextBox**.**Focus**();**

**}**

**}**

//method to check bagel quantity of that particular array location

**private** void CheckBagelQuantity**(object** sender**,** EventArgs e**)**

**{**

**if** **((**bagelTypeListBox**.**SelectedIndex **!=** **-**1**)** **&&** **(**bagelSizeListBox**.**SelectedIndex **!=** **-**1**))**

**{**

//add the index and add the quantity of that index to a integer variable

bagelTypeIndex **=** bagelTypeListBox**.**SelectedIndex**;**

bagelSizeIndex **=** bagelSizeListBox**.**SelectedIndex**;**

bagelQuantity **=** tempBagelStock**[**bagelTypeIndex**,** bagelSizeIndex**];**

**}**

**}**

//method to caluclate bagel price

**private** void CalculateBagelPrice**(object** sender**,** EventArgs e**)**

**{**

**try**

**{**

//call this method

ChosenBagelPrice**(**sender**,** e**);**

decimal priceOfEachBagel**;**

decimal price**;**

//enter quantity of bagels to choose

enterBagelQuantity **=** bagelQuantityTextBox**.**Text**;**

//regex to check the type of numbers

Match bagelInputMatch **=** Regex**.**Match**(**enterBagelQuantity**,** "^[1-9]"**);**

//checks the input type

**if** **(**bagelInputMatch**.**Success **==** **true)**

**{**

//calculates the price after entering user quantity

quantityOfBagel **=** int**.**Parse**(**enterBagelQuantity**);**

priceOfEachBagel **=** decimal**.**Parse**((**bagelPriceTextBox**.**Text**),** NumberStyles**.**Currency**);**

price **=** **(**quantityOfBagel **\*** priceOfEachBagel**);**

bagelTotalPriceTextbox**.**Text **=** price**.**ToString**(**"c"**);**

**}**

**else**

**{**

//validates to enter in a correct format

MessageBox**.**Show**(**"Enter quantity in correct format"**,** "Invalid input error"**);**

bagelQuantityTextBox**.**Clear**();**

**}**

**}**

**catch**

**{**

//catch to handle to and ask user to enter a value

bagelQuantityTextBox**.**Clear**();**

bagelTotalPriceTextbox**.**Text **=** ""**;**

MessageBox**.**Show**(**"Enter a value"**,** "Enter Quantity"**);**

**}**

**}**

//event handler after clicking the order button

**private** void AddOrder\_Button\_Click**(object** sender**,** EventArgs e**)**

**{**

//checks the entered quantity if its is lesser than the stock

**if** **(**quantityOfBagel **<=** tempBagelStock**[**bagelTypeIndex**,** bagelSizeIndex**])**

**{**

//checks if items are selected in the list boxes

**if** **(**bagelTypeListBox**.**SelectedIndex **==** **-**1**)**

**{**

MessageBox**.**Show**(**"You have not selected a Bagel type!! Choose a bagel Type"**,** "No Bagel selected"**);**

**}**

**else** **if** **(**bagelSizeListBox**.**SelectedIndex **==** **-**1**)**

**{**

MessageBox**.**Show**(**"You have not selected a Bagel size!!Choose a bagel size"**,** "No Bagel size selected"**);**

**}**

//checks if there is a quantity entered

**else** **if** **(**bagelQuantityTextBox**.**Text **==** ""**)**

**{**

MessageBox**.**Show**(**"Enter the Quantity of Bagels"**,** "No Bagels entered"**);**

bagelQuantityTextBox**.**Focus**();**

**}**

**else**

**{**

//after successful validation following activities are performed to add the order selected by the

//user

CheckBagelQuantity**(**sender**,** e**);**

CheckBagelStock**(**sender**,** e**);**

orderedBagelTypesListBox**.**Items**.**Add**(**bagelTypeTextBox**.**Text**);**

orderedBagelSizeListBox**.**Items**.**Add**(**bagelSizeTextBox**.**Text**);**

orderedBagelQuantityListBox**.**Items**.**Add**(**bagelQuantityTextBox**.**Text**);**

orderPriceListBox**.**Items**.**Add**(**bagelTotalPriceTextbox**.**Text**);**

confirmOrderbutton**.**Enabled **=** **true;**

cancelOrderButton**.**Enabled **=** **true;**

//clear fields to facilitate the new order

ClearFields**(**sender**,** e**);**

**}**

**}**

//message to pop up insufficient quantity

**else** **if** **(**tempBagelStock**[**bagelTypeIndex**,** bagelSizeIndex**]** **==** 0**)**

**{**

MessageBox**.**Show**(**"There are no stock available for this bagel type, Please choose another item"**,** "No stock available"**);**

ClearFields**(**sender**,** e**);**

**}**

//asks user if he wants to choose available quantity

**else** **if** **(**quantityOfBagel **>** tempBagelStock**[**bagelTypeIndex**,** bagelSizeIndex**])**

**{**

DialogResult okresult**;**

okresult **=** MessageBox**.**Show**(**"There are " **+** tempBagelStock**[**bagelTypeIndex**,** bagelSizeIndex**]** **+** " stocks available. would you" **+**

" like to add " **+** tempBagelStock**[**bagelTypeIndex**,** bagelSizeIndex**]** **+** " bagels?"**,** "No enough stocks!!"**,** MessageBoxButtons**.**OKCancel**);**

**if** **(**okresult **==** DialogResult**.**OK**)**

**{**

//if ok then that quantity is selected

bagelQuantityTextBox**.**Text **=** tempBagelStock**[**bagelTypeIndex**,** bagelSizeIndex**].**ToString**();**

**}**

//else the fields are cleared

**else**

**{**

ClearFields**(**sender**,** e**);**

**}**

**}**

**}**

//calling the methods on change in selection in the list box

**private** void BagelSizeListBox\_SelectedIndexChanged**(object** sender**,** EventArgs e**)**

**{**

**if** **(**bagelQuantityTextBox**.**Text **!=** ""**)**

**{**

CalculateBagelPrice**(**sender**,** e**);**

**}**

**else**

**{**

ChosenBagelPrice**(**sender**,** e**);**

**}**

**}**

//calling the methods on change in selection in the list box

**private** void BagelTypeListBox\_SelectedIndexChanged**(object** sender**,** EventArgs e**)**

**{**

**if** **(**bagelQuantityTextBox**.**Text **!=** ""**)**

**{**

CalculateBagelPrice**(**sender**,** e**);**

**}**

**else**

**{**

ChosenBagelPrice**(**sender**,** e**);**

**}**

**}**

//calling the methods on change in selection in the text box

**private** void BagelQuantityTextBox\_TextChanged**(object** sender**,** EventArgs e**)**

**{**

**if** **(**bagelQuantityTextBox**.**Text **==** ""**)**

**{**

bagelTotalPriceTextbox**.**Text **=** ""**;**

**}**

**else**

**{**

CalculateBagelPrice**(**sender**,** e**);**

**}**

**}**

//calling methods on form load

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

**{**

**try**

**{**

CallBagelstockFile**(**sender**,** e**);**

CallTempBagelStockFile**(**sender**,** e**);**

CallBagelPriceFile**(**sender**,** e**);**

orderedBagelTypesListBox**.**Items**.**Clear**();**

orderedBagelSizeListBox**.**Items**.**Clear**();**

orderPriceListBox**.**Items**.**Clear**();**

**}**

**catch** **(**Exception**)**

**{**

MessageBox**.**Show**(**"Error on form load"**);**

**}**

**}**

//calling the call bagel stock file method to load the array from the file

**private** void CallBagelstockFile**(object** sender**,** EventArgs e**)**

**{**

**try**

**{**

//read the text in the location

String stockFileRead **=** File**.**ReadAllText**(**bagelStockFileLocation**);**

//replcat with ""

stockFileRead**.**Replace**(**"\r"**,** ""**);**

int i **=** 0**,** j **=** 0**;**

//for loop to load the array from the file

**foreach** **(**var rowRead **in** stockFileRead**.**Split**(**'\n'**))**

**{**

j **=** 0**;**

**foreach** **(**var ColRead **in** rowRead**.**Trim**().**Split**(**','**))**

**{**

bagelStock**[**i**,** j**]** **=** int**.**Parse**(**ColRead**.**Trim**());**

j**++;**

**}**

i**++;**

**}**

**}**

**catch** **(**Exception**)**

**{**

MessageBox**.**Show**(**"File not found!!"**);**

**this.**Close**();**

**}**

**}**

//load the contents of stock array to a temporary array

**private** void CallTempBagelStockFile**(object** sender**,** EventArgs e**)**

**{**

**try**

**{**

**for** **(**int j **=** 0**;** j **<** rowOfBagels**;** j**++)**

**{**

**for** **(**int k **=** 0**;** k **<** columnOfBagels**;** k**++)**

**{**

tempBagelStock**[**j**,** k**]** **=** bagelStock**[**j**,** k**];**

**}**

**}**

**}**

**catch** **(**Exception**)**

**{**

MessageBox**.**Show**(**"Error in temporary Bagel Stock"**);**

**}**

**}**

//call Bagel price file method from the file to load into an array

**private** void CallBagelPriceFile**(object** sender**,** EventArgs e**)**

**{**

**try**

**{**

String stockPriceRead **=** File**.**ReadAllText**(**bagelPriceFileLocation**);**

stockPriceRead**.**Replace**(**"\r"**,** ""**);**

int i **=** 0**,** j **=** 0**;**

**foreach** **(**var rowRead **in** stockPriceRead**.**Split**(**'\n'**))**

**{**

j **=** 0**;**

**foreach** **(**var ColRead **in** rowRead**.**Trim**().**Split**(**','**))**

**{**

bagelPrice**[**i**,** j**]** **=** decimal**.**Parse**(**ColRead**.**Trim**());**

j**++;**

**}**

i**++;**

**}**

**}**

**catch**

**{**

MessageBox**.**Show**(**"File not found!!"**);**

**this.**Close**();**

**}**

**}**

//cancel order button

**private** void CancelOrderButton\_Click**(object** sender**,** EventArgs e**)**

**{**

DialogResult okResult**;**

okResult **=** MessageBox**.**Show**(**"Do you want to cancel the order?"**,** "Order Cancellation"**,** MessageBoxButtons**.**OKCancel**);**

**if** **(**okResult **==** DialogResult**.**OK**)**

**{**

//clear the listboxes and call the temporary bagel stock method

orderedBagelTypesListBox**.**Items**.**Clear**();**

orderedBagelSizeListBox**.**Items**.**Clear**();**

orderedBagelQuantityListBox**.**Items**.**Clear**();**

orderPriceListBox**.**Items**.**Clear**();**

CallTempBagelStockFile**(**sender**,** e**);**

cancelOrderButton**.**Enabled **=** **false;**

confirmOrderbutton**.**Enabled **=** **false;**

**}**

**else**

**{**

cancelOrderButton**.**Enabled **=** **false;**

**}**

**}**

//clear fields method to clear the fields

**private** void ClearFields**(object** sender**,** EventArgs e**)**

**{**

bagelTypeListBox**.**ClearSelected**();**

bagelSizeListBox**.**ClearSelected**();**

bagelTypeTextBox**.**Clear**();**

bagelSizeTextBox**.**Clear**();**

bagelPriceTextBox**.**Clear**();**

bagelQuantityTextBox**.**Clear**();**

bagelQuantityTextBox**.**Enabled **=** **false;**

bagelTotalPriceTextbox**.**Clear**();**

**}**

//event handler for confirm button

**private** void ConfirmOrderbutton\_Click**(object** sender**,** EventArgs e**)**

**{**

**try**

**{**

//read the values from the list boxes

string bagelTypeRead **=** ""**;**

**foreach** **(**string readString **in** orderedBagelTypesListBox**.**Items**)**

**{**

bagelTypeRead **+=** readString**.**ToString**()** **+** "\n"**;**

**}**

string bagelSizeRead **=** ""**;**

**foreach** **(**string readstring **in** orderedBagelSizeListBox**.**Items**)**

**{**

bagelSizeRead **+=** readstring**.**ToString**()** **+** "\n"**;**

**}**

int bagelsSold **=** 0**;** **;**

string bagelQuantityRead **=** ""**;**

**foreach** **(**string readstring **in** orderedBagelQuantityListBox**.**Items**)**

**{**

bagelQuantityRead **+=** readstring**.**ToString**()** **+** "\n"**;**

bagelsSold **+=** int**.**Parse**(**readstring**);**

**}**

//calculate the value per transaction

string bagelPriceRead **=** ""**;**

decimal totalBagelPricePerTransac **=** 0**;**

**foreach** **(**string readstring **in** orderPriceListBox**.**Items**)**

**{**

bagelPriceRead **+=** readstring**.**ToString**()** **+** "\n"**;**

totalBagelPricePerTransac **+=** decimal**.**Parse**(**readstring**,** NumberStyles**.**Currency**);**

**}**

//increase the transaction count by 1

transactionNumber **+=** 1**;**

DialogResult okresult**;**

//message box to display the confirmed order

okresult **=** MessageBox**.**Show**(**"Transaction Number:\n" **+** transactionNumber **+** "\n Bagels chosen by the customer: \n" **+** bagelTypeRead **+** "\n Bagel size chosen by the customer: \n"

**+** bagelSizeRead **+** "\n Bagel Quantity:\n" **+** bagelQuantityRead

**+** "\n Bagel Price: \n" **+** bagelPriceRead **+** "\n Total Price \n" **+** totalBagelPricePerTransac**.**ToString**(**"c"**),** "Do You want to confirm your order"**,** MessageBoxButtons**.**OKCancel**);**

//if the result is ok confirm order and update the array and write it to a file and

//cleat the fields

**if** **(**okresult **==** DialogResult**.**OK**)**

**{**

MessageBox**.**Show**(**"Your order has been confirmed"**,** "Order confirmation"**);**

ClearFields**(**sender**,** e**);**

orderedBagelTypesListBox**.**Items**.**Clear**();**

orderedBagelSizeListBox**.**Items**.**Clear**();**

orderedBagelQuantityListBox**.**Items**.**Clear**();**

orderPriceListBox**.**Items**.**Clear**();**

cancelOrderButton**.**Enabled **=** **false;**

confirmOrderbutton**.**Enabled **=** **false;**

WriteFileAfterConfirm**(**sender**,** e**);**

//calculate the summary for future reference for the day

totalTransactionCount **=** transactionNumber**;**

totalBagelsSold **+=** bagelsSold**;**

totalValue **+=** totalBagelPricePerTransac**;**

averageTransactions **=** **(**totalValue**)** **/** totalTransactionCount**;**

**}**

**else**

**{**

//else the decrease the transaction number

transactionNumber **-=** 1**;**

ClearFields**(**sender**,** e**);**

cancelOrderButton**.**Enabled **=** **true;** **}**

**}**

**catch** **(**Exception**)**

**{**

MessageBox**.**Show**(**"Order Confirmation Error"**,** "Error in confirmation"**);**

**}**

**}**

//check the bagel stock method and reduce it at that array location

**private** void CheckBagelStock**(object** sender**,** EventArgs e**)**

**{**

int subtractStockValue **=** **(**tempBagelStock**[**bagelTypeIndex**,** bagelSizeIndex**]** **-** quantityOfBagel**);**

tempBagelStock**[**bagelTypeIndex**,** bagelSizeIndex**]** **=** subtractStockValue**;**

**}**

//event handler for summary button click to calucalte summary

**private** void SummaryButton\_Click**(object** sender**,** EventArgs e**)**

**{**

**if(**totalTransactionCount**!=**0**)**

**{**

//display all the values in the summary panel calculated and stored after summary

transactionCountSummaryTextBox**.**Text **=** totalTransactionCount**.**ToString**();**

bagelsSoldTextBox**.**Text **=** totalBagelsSold**.**ToString**();**

totalTransactionTextBox**.**Text **=** totalValue**.**ToString**();**

averageTransactionsTextBox**.**Text **=** averageTransactions**.**ToString**();**

**}**

**else**

**{**

MessageBox**.**Show**(**"No Transactions to display for today!!"**,**"No transactions for today!"**);**

**}**

**}**

//clear button event to clear fields and call the temporary array to replace the old values

**private** void ClearButton\_Click**(object** sender**,** EventArgs e**)**

**{**

transactionCountSummaryTextBox**.**Clear**();**

bagelsSoldTextBox**.**Clear**();**

totalTransactionTextBox**.**Clear**();**

averageTransactionsTextBox**.**Clear**();**

**}**

//display the management report on click of this button at any point of time

**private** void ManagementReportButton\_Click**(object** sender**,** EventArgs e**)**

**{**

**try**

**{**

//stream writer to write the file

StreamWriter managementReportWriter**;**

//create a new file at that location each time and add new text

managementReportWriter**=**File**.**CreateText**(**managementReportFileLocation**);**

//display format

managementReportWriter**.**WriteLine**(**"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Mr.Bagel Kiltimagh \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**);**

managementReportWriter**.**WriteLine**(**"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Stock Report \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**);**

managementReportWriter**.**WriteLine**(**"\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" **+** DateTime**.**Now**.**ToString**(**"hh:mm dd/MM/yyyy"**)** **+** "\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**);**

managementReportWriter**.**WriteLine**(**"TYPE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_SIZE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_QUANTITY"**);**

//loop through each bagel type array while the nested for loop loops through

//the bagel size array and displays that array location

**for** **(**int i **=** 0**;** i **<** bagelTypeListBox**.**Items**.**Count**;** i**++)**

**{**

**for** **(**int j **=** 0**;** j **<** bagelSizeListBox**.**Items**.**Count**;** j**++)**

**{**

managementReportWriter**.**Write**(**bagelTypeListBox**.**Items**[**i**]** **+** "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" **+** bagelSizeListBox**.**Items**[**j**]** **+** "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" **+** tempBagelStock**[**i**,** j**]** **+** Environment**.**NewLine**);**

**}**

**}**

managementReportWriter**.**Close**();**

**}**

**catch**

**{**

MessageBox**.**Show**(**"file Write exception"**);**

**}**

//display the file in the interface on click of button

Process**.**Start**(**managementReportFileLocation**);**

**}**

//closes the application

**private** void ExitButton\_Click**(object** sender**,** EventArgs e**)**

**{**

**this.**Close**();**

**}**

//write the file after click of confirm button

**private** void WriteFileAfterConfirm**(object** sender**,** EventArgs e**)**

**{**

//create the stream write object and write it at that file location

System**.**IO**.**StreamWriter streamWriter **=** **new** System**.**IO**.**StreamWriter**(**bagelStockFileLocation**);**

**for** **(**int i **=** 0**;** i **<** rowOfBagels**;** i**++)**

**{**

**for** **(**int j **=** 0**;** j **<**columnOfBagels**;** j**++)**

**{**

//separate each array with a " ,"

string spacing **=** **(**j **==** columnOfBagels **-** 1**)** **?** "" **:** ","**;**

streamWriter**.**Write**(**tempBagelStock**[**i**,** j**]+**spacing**);**

**}**

//if the i is equal to 4 or encounters "" then go to new line

string nextLine **=** **(**i **==** rowOfBagels**-**1**)** **?** "" **:** Environment**.**NewLine**;**

streamWriter**.**Write**(**nextLine**);**

**}**

//close the stream writer object

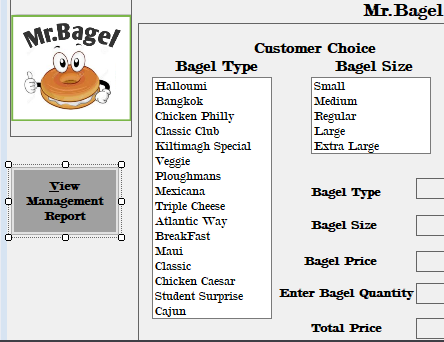
streamWriter**.**Close**();**

**}**

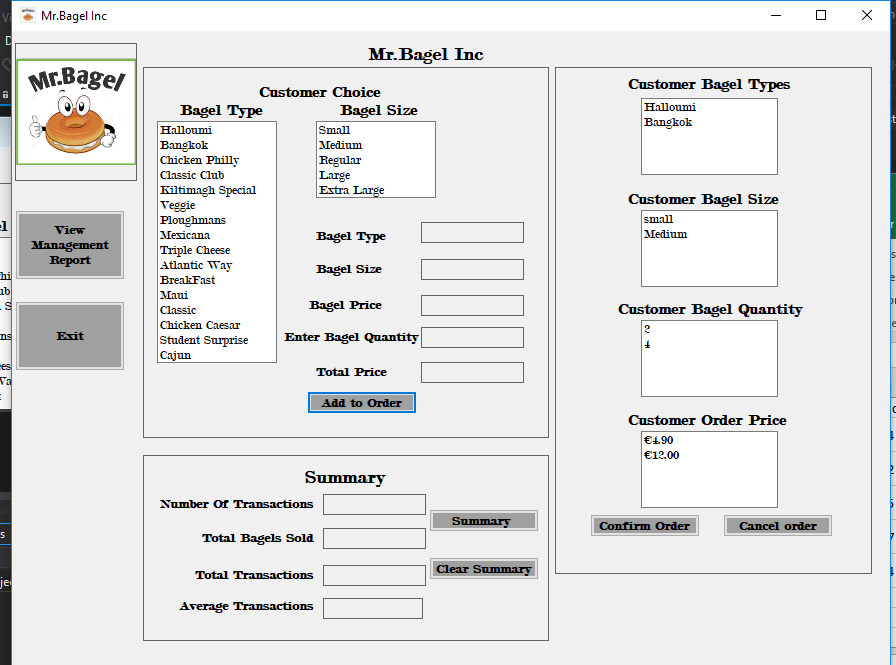
**}**

**}**

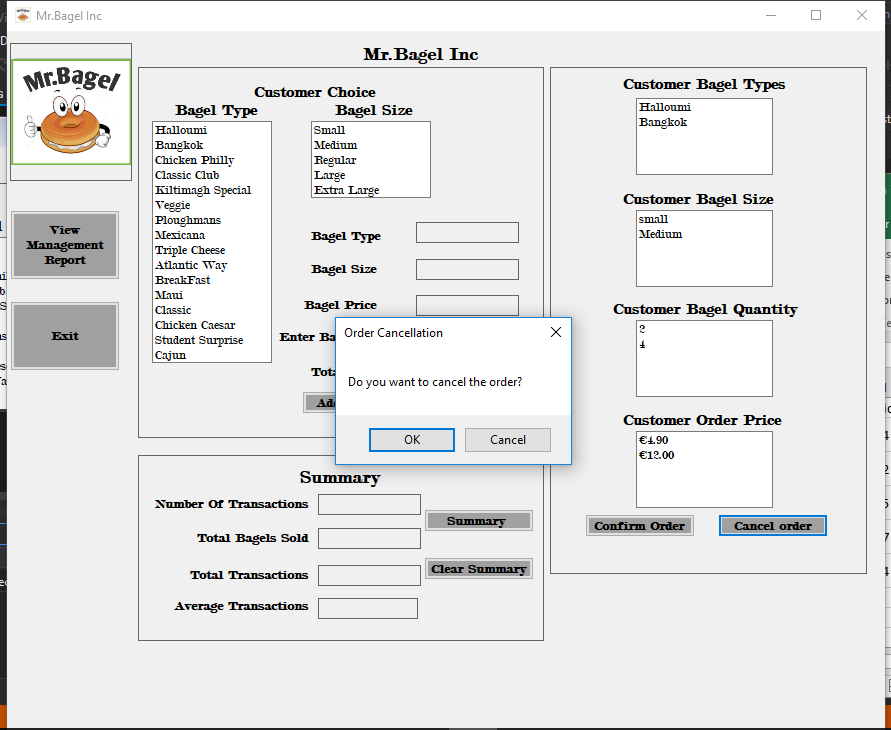
**FIGURES**



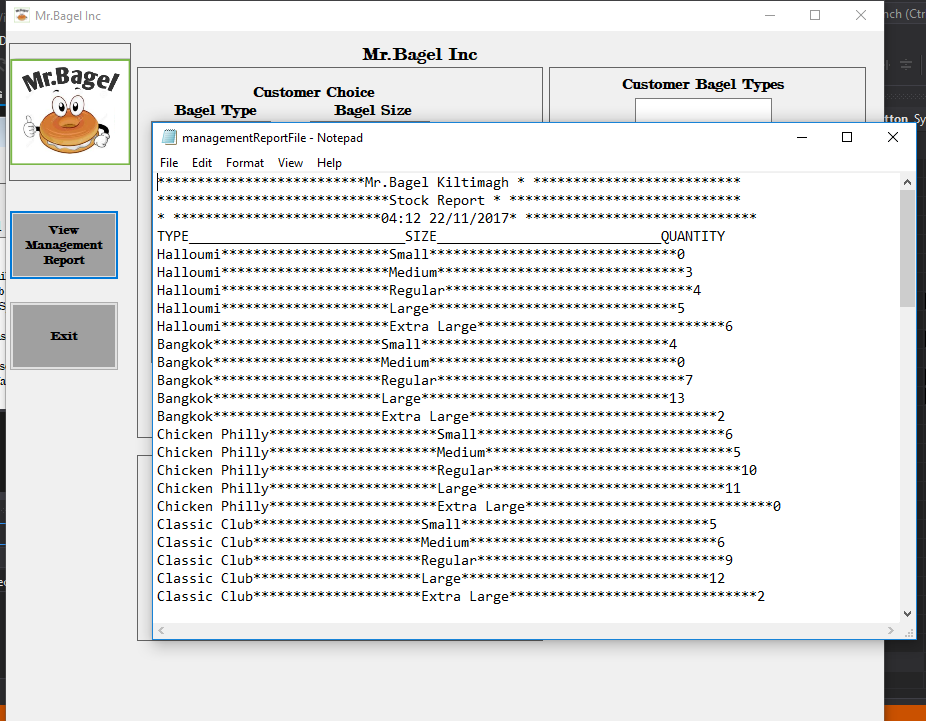
**FIG1: Access Keys**



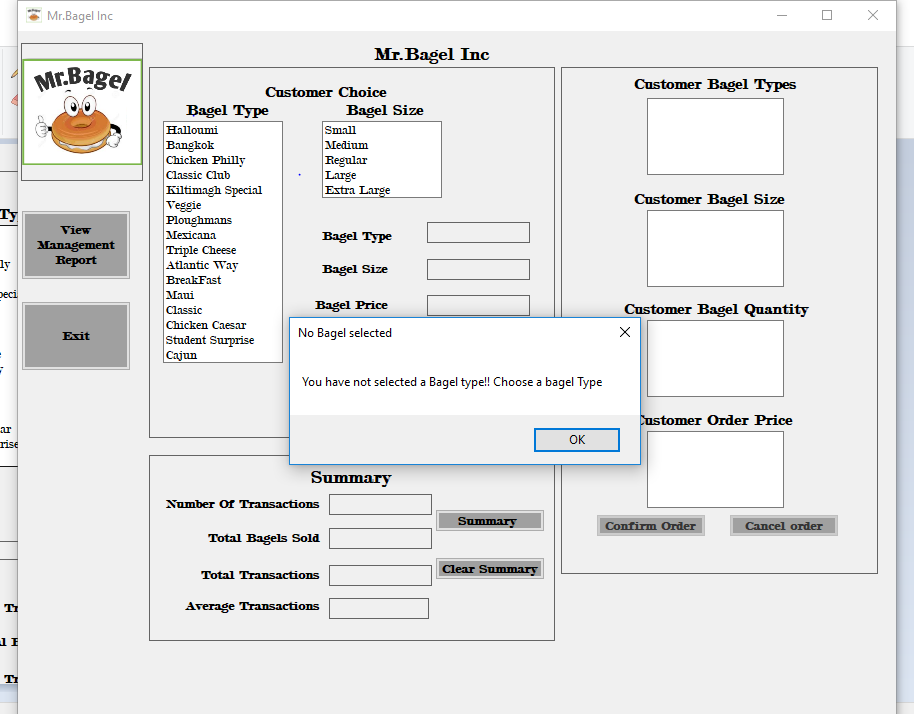
**Fig 2: Adding order**



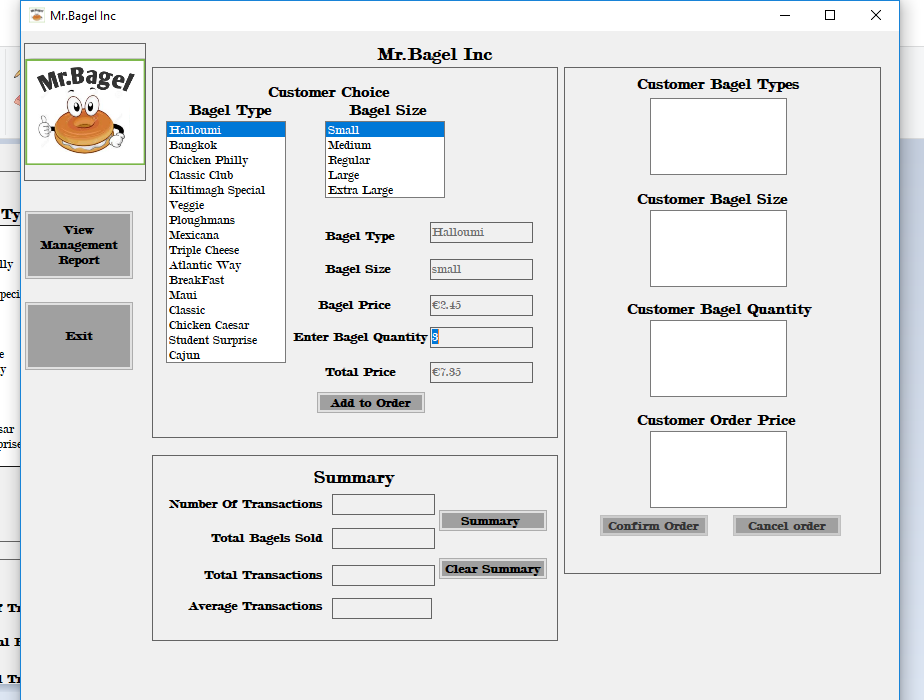
**Fig 3: cancel order confirmation**



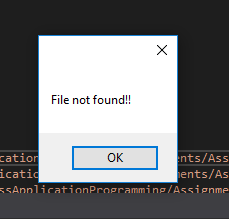
**Fig 4: Check management report any time on the interface**



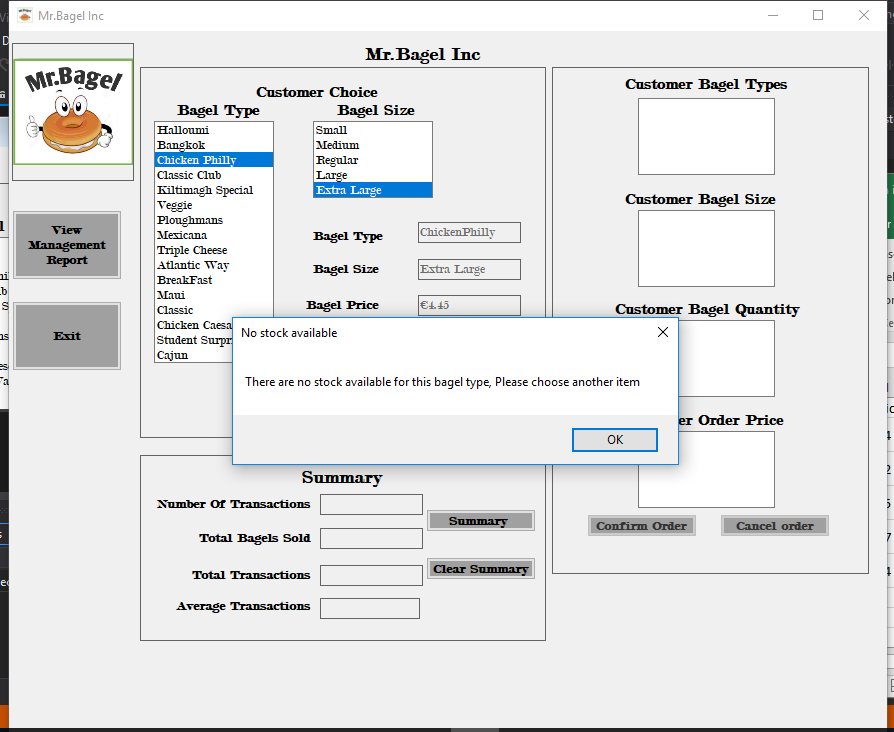
**Fig5:click on order without selecting bagel type**



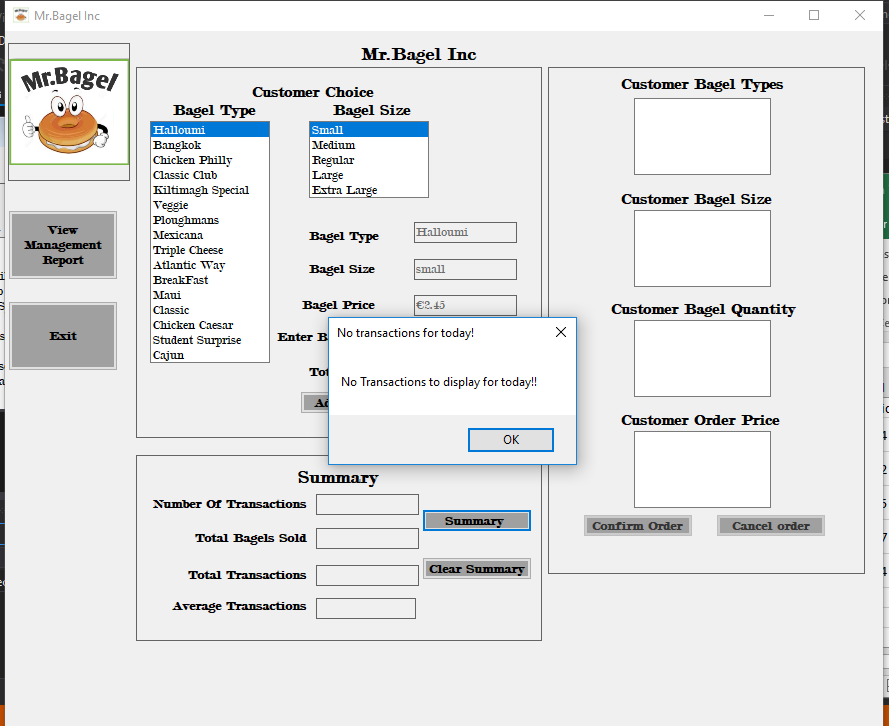
**Fig 6: Entering correct quantity**



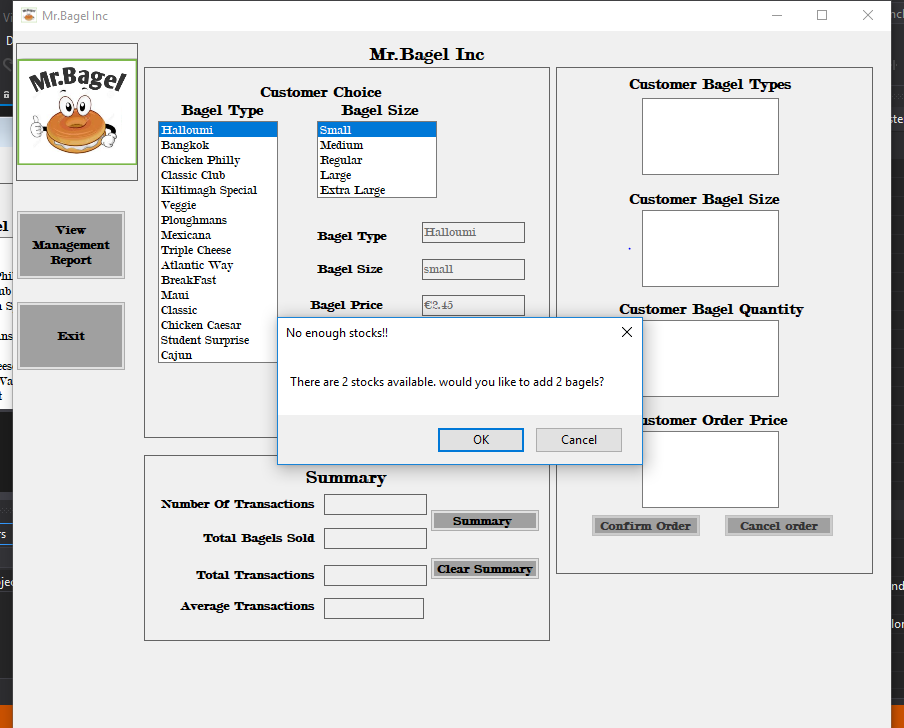
**Fig 7: File not found error**



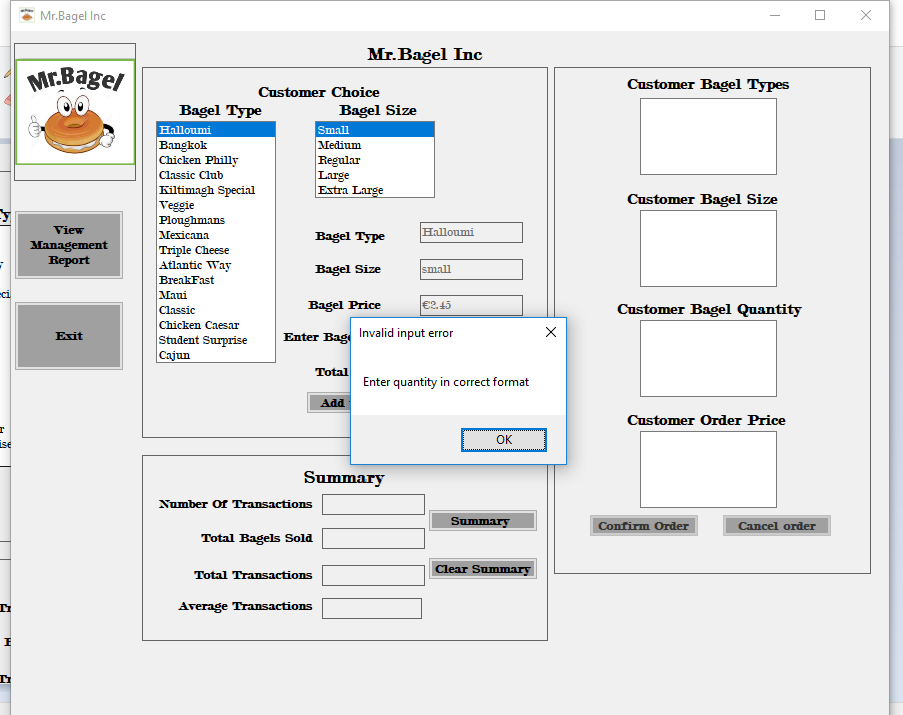
**Fig 8: No stock available**



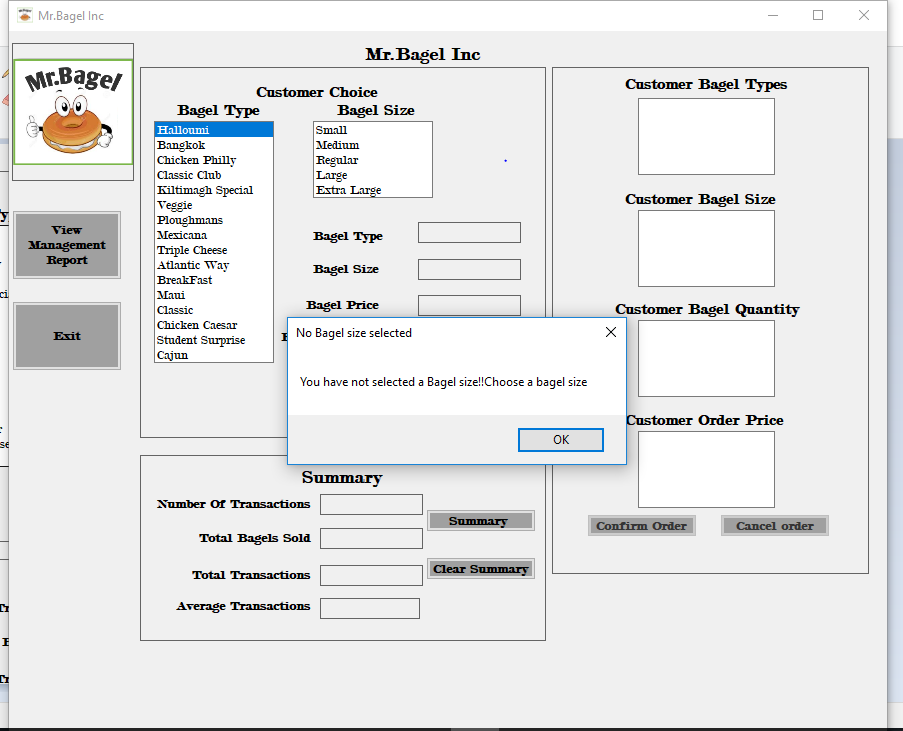
**Fig 9: No summary to display for the day**



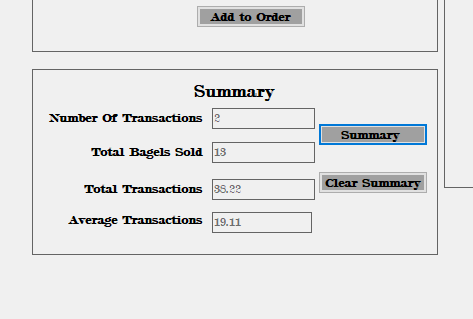
**Fig 10: Enter quantity available**



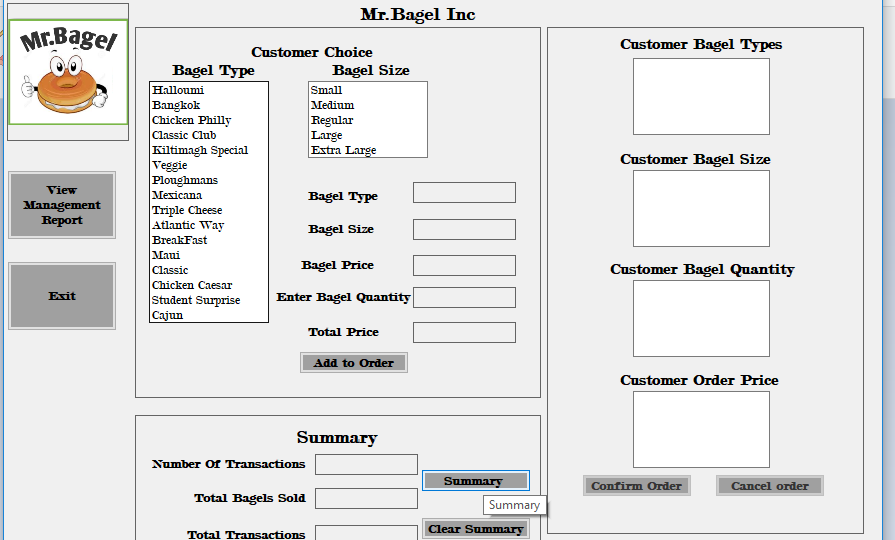
**Fig 11: wrong quantity format**



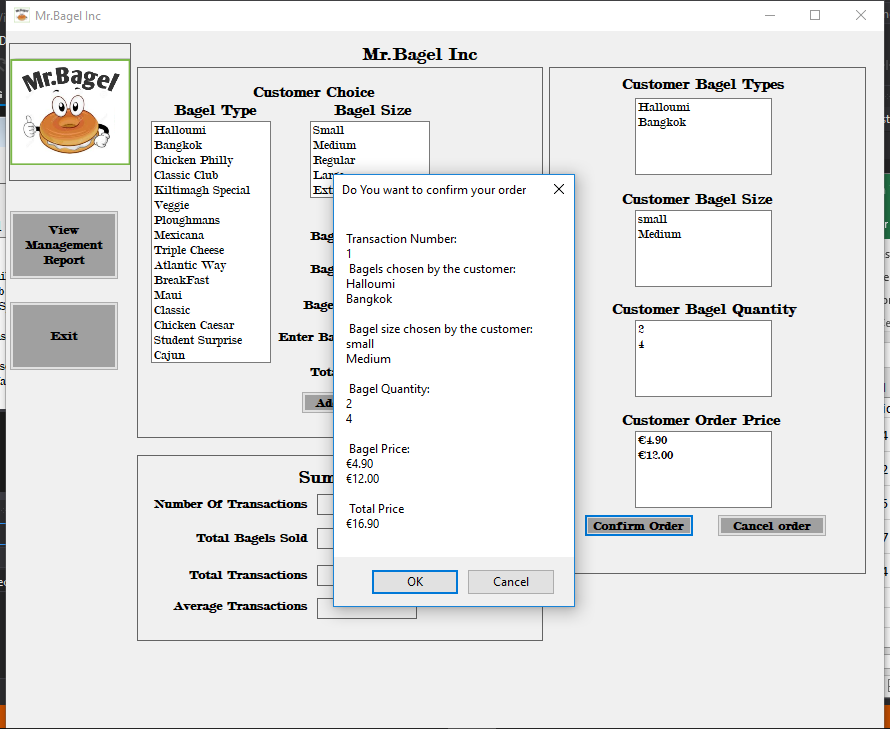
**Fig 12: when order clicked before entering bagel size**



**Fig 13: Summary event**



**Fig 14: ToolTip**



**Fig 15:Transaction Confirmation**