MBA 614

VBA Final Project

Eli H. Trejo

"Recipe Organizer Application"

Table of Contents

RECIPE ORGANIZER

Executive summary

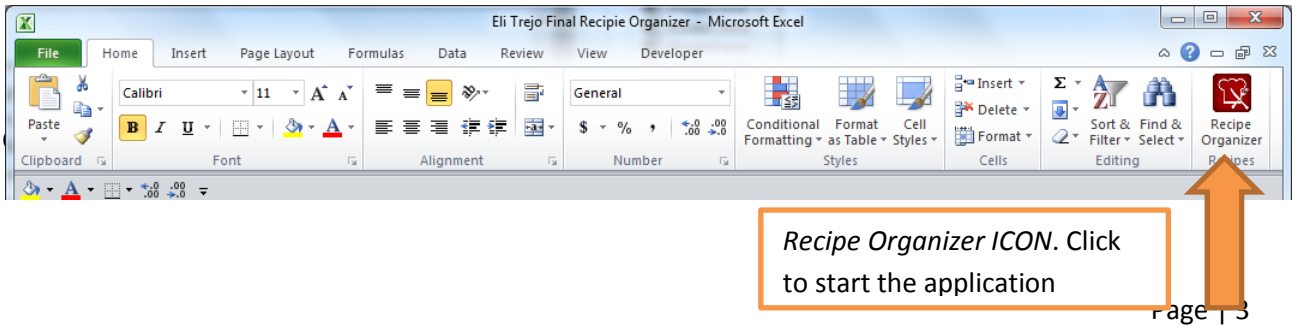
Description of the business: the application was designed for family use. Currently, my wife, as well as most of my family members, keeps her recipes in 3" by 5" flashcards. When a flashcard is damaged the recipe is lost, when a new recipe is received or when a recipe is shared, several minutes are wasted transcribing the recipe by hand.

Overview of the system: This application allows the user to neatly organize his/her food recipes more effectively than existing free web applications. Besides being able to enter, modify, and delete recipes in a matter of seconds, the user has both simple and advanced search capabilities to navigate to recipes stored in his/her local computer as well as recipes from www.allprecipes.com. Assuming than an internet connection is available, the application allows the user to import recipes from www.allprecipes.com to his/her local spreadsheet with a simple click. Also, the application allows the user to email recipes (imported from the web or entered manually) to one or multiple users in a matter of seconds.

The application allows the user to do all the necessary recipe transactions without having to directly navigate in Excel (everything is done through forms). If in the feature the user decides to move all of his/her stored recipes into a database, such as MS Access, there will be no need to modify the data stored in Excel. All the recipes are broken down into different parts (tables) and each part is stored in different sheets to follow best database design conventions. In addition, all the restrictions in addition and deletion of data common to a database are being implemented in this application to protect data integrity. Regardless of the database of use, the user will be able to import all its recipes into the desired database tables without any modifications. Overall, this project is a complete application of all the concepts and tools learned in class as well as from other VBA teachings sources.

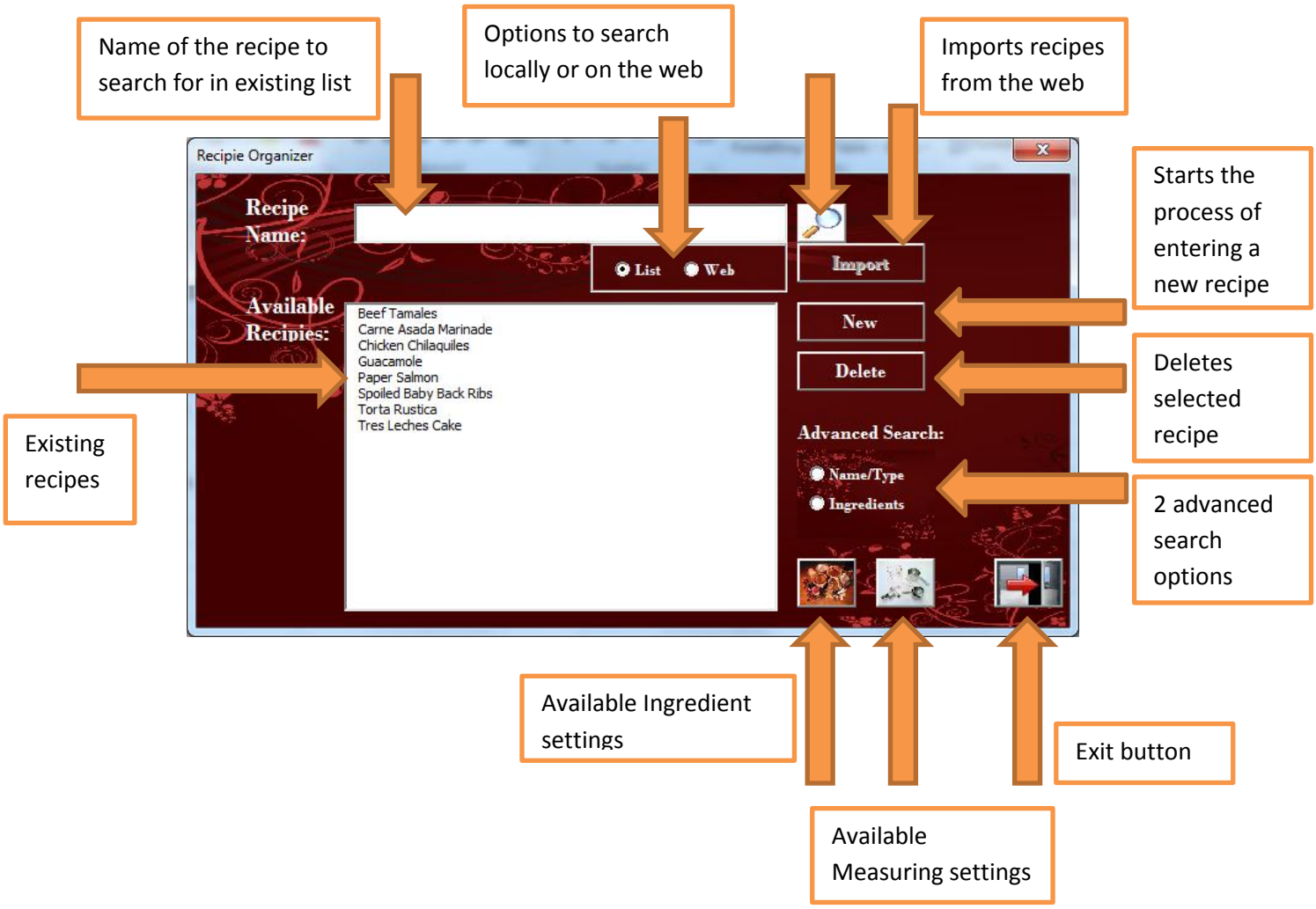
Implementation Documentation

Start the application: to start the application the use needs to be in the "Home Excel tab and click on the far right ICON named "Recipe Organizer." As soon as the button is clicked the application will start and will do all the desired user transactions.



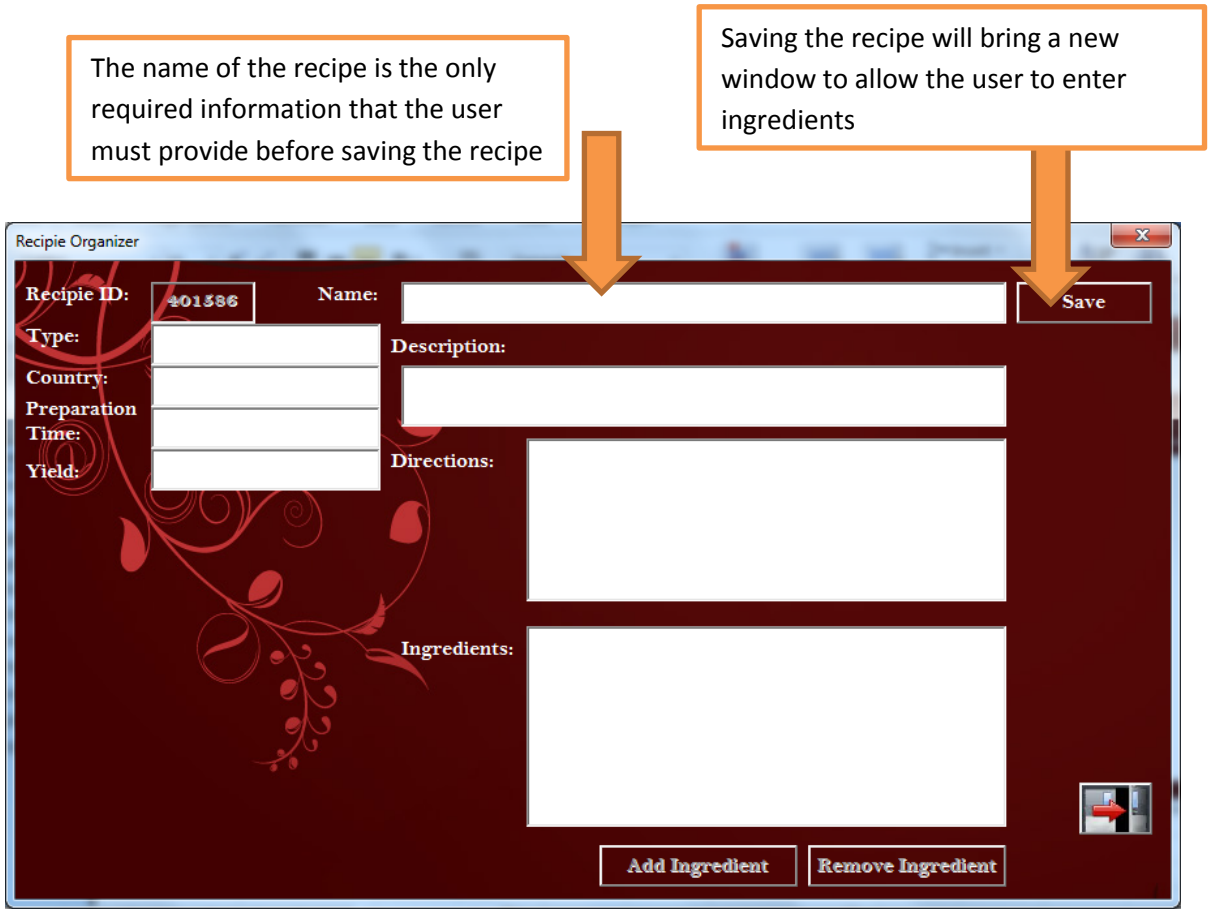
Recipe Organizer ICON. Click to start the application

Basic description of the welcoming screen: the screen below is what we can consider the welcoming. It will serve as the basic navigation window and will allow the use to start the process of adding, editing, and deleting recipes.



Add a new recipe manually: To start the process of creating a new recipe the user must press the "New" button on the welcoming screen. Once the user clicks the "new" recipe button the application will open a new form where the user will have the opportunity to enter the basic

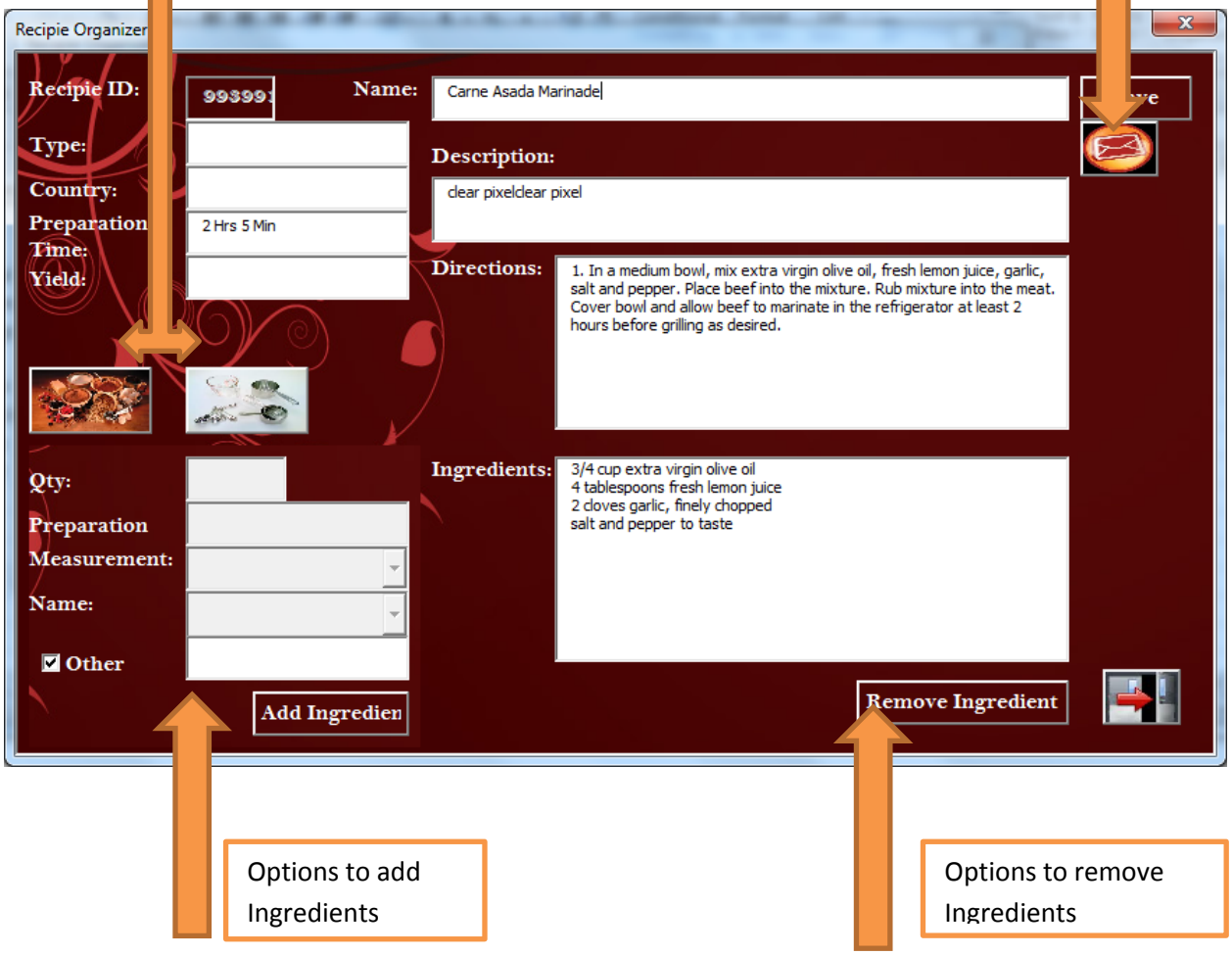
information of a recipe. The window below automatically generates a recipe id, which might be irrelevant for the user. The user is required to enter at least the name of the recipe before saving the recipe. Once the user saves the recipe a new window will appear where the user will be allowed to enter the ingredients of the recipe.



As shown in the previous form, on the first stage of creating a recipe the add and remove ingredients buttons are blocked. Once the user save the recipe, the current form disappears and a new one appears (see below). On the new window not only the user has the chance to add ingredients but it can modify them along with the rest of the recipe. Also, at this point the user already has the tools to email the recipe to the recipient of his choice.

If needed the user can adjust the ingredients and measurement settings

User can email the recently entered recipe to the recipients of his/her choice



Options to add Ingredients

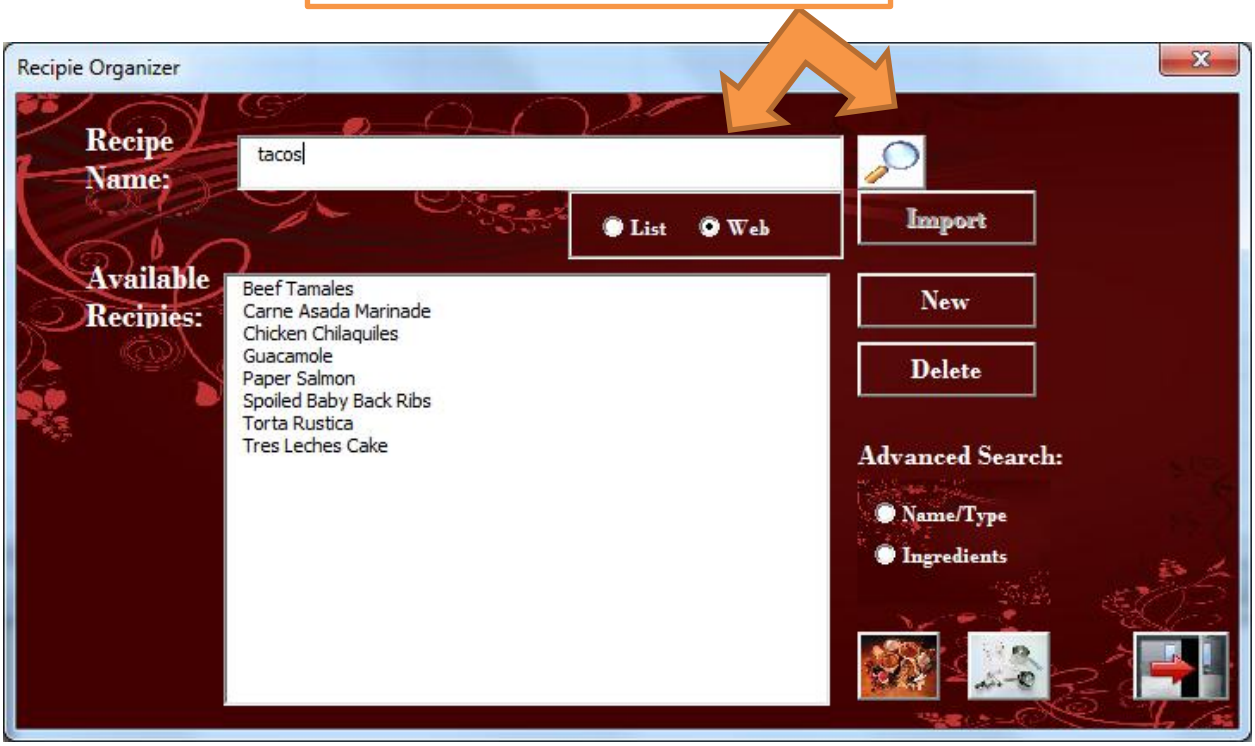
Options to remove Ingredients

While adding ingredients to a recipe the user has the choice to select ingredients and measurements from the combo boxes or the user can opt for entering its own data. The images show how the user can lock or unlock the combo boxes by selecting the “Other” check box.

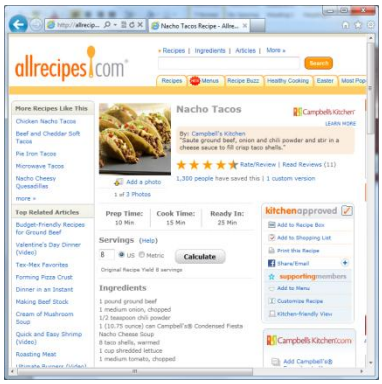
Add a new recipe from the web: by default the import button is unavailable. To activate the button the user needs to type the name of the recipe that he/she is looking for. Then the "web" option button must be selected and the search button must be clicked.

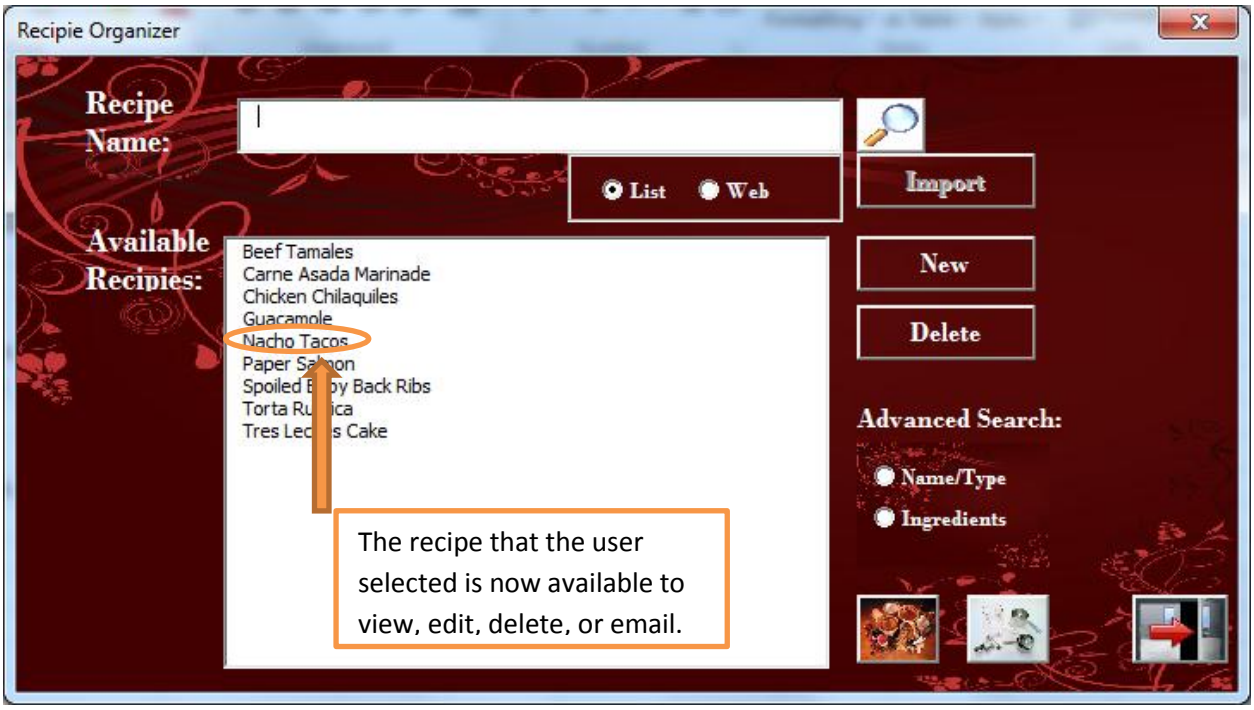
Internet Explorer will be launched and www.allrecipes.com will display all the possible matches for the recipe that the user is looking for. Once the user clicks on the recipe of his/her choice, the import button will be enable and by clicking it the user will be able to import all the information from the web to the excel spreadsheets.

The user enters a recipe name, selects web, and clicks on the search button

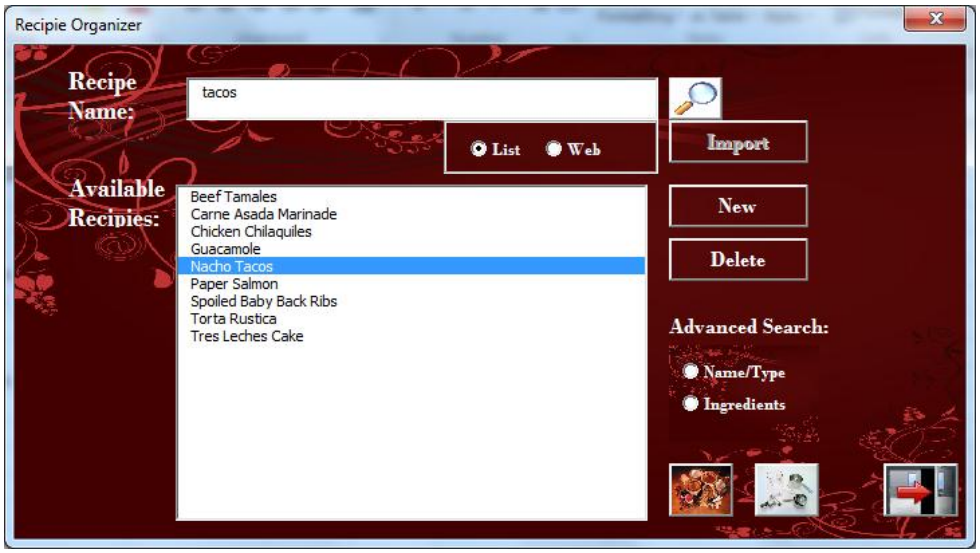


The left image shows the results of the search that the user requested. Once the user selects one of the display recipes (see right image), the import button becomes enable and the import can be executed.



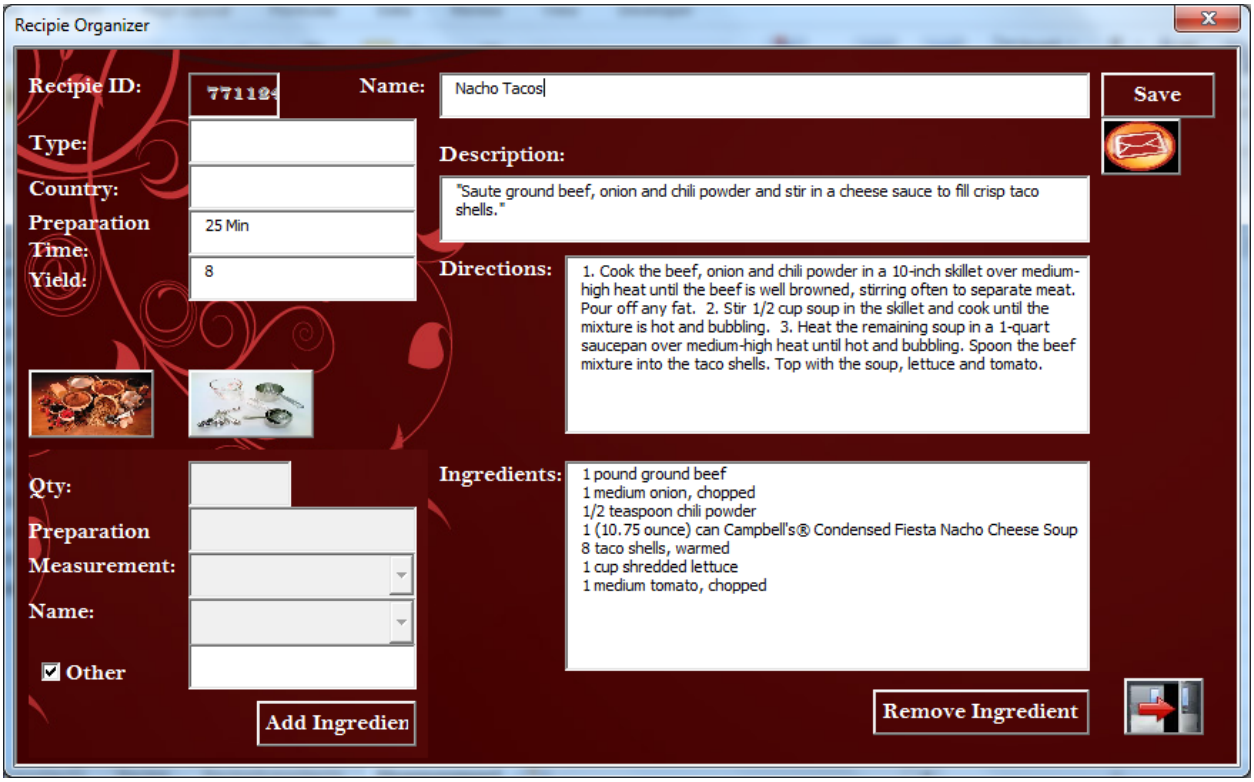


Edit and existing recipe: the user must select an existing recipe from the available recipes list and double click. If the list is too long the user can do a basic search where he/she can type part of the name of the recipe and do a search from list.

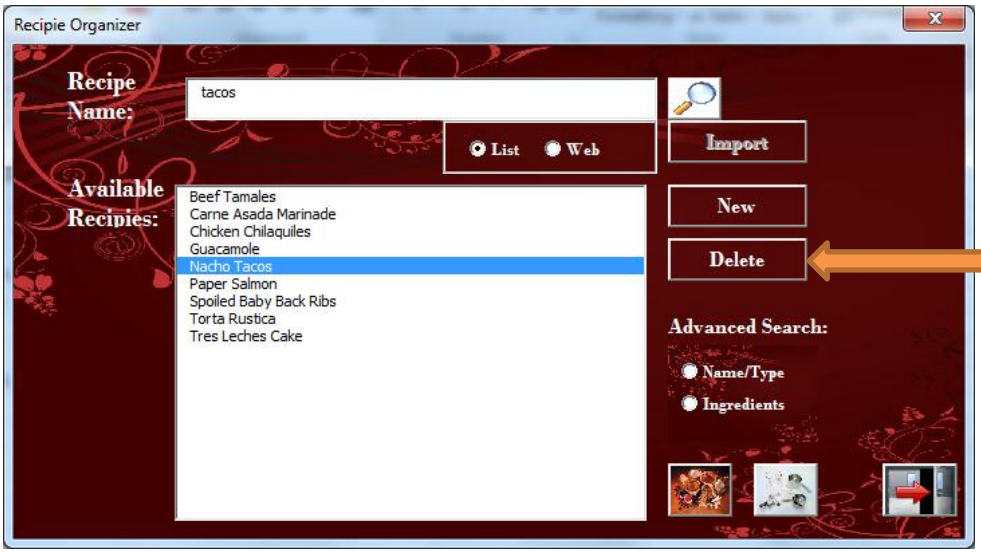


Once the user specifies the name of a recipe and executes a basic search the application finds it on the list. By double clicking the edit window appears

This is the recipe that the user imported from the web. This is the edit window where the user has the freedom to modify or just view the recipe.



Delete an existing recipe: similar to other transactions, the user will have to select the recipe that needs to be removed from the recipe organizer. Once the recipe is selected on the welcoming screen, the user has to click on the delete button. The application will completely remove the recipe and all its pertaining data from the system.



Search for an existing recipe: as previously indicated, the use can do a simple search on the main textbox to find recipes by their complete or partial name.

From web: If the user wants to search on the web the user just needs to type the name or partial name of the recipe, select web, and click on the search

Locally: If the user wants to search on the web the user just needs to type the name or partial name of the recipe, select list, and click on the search

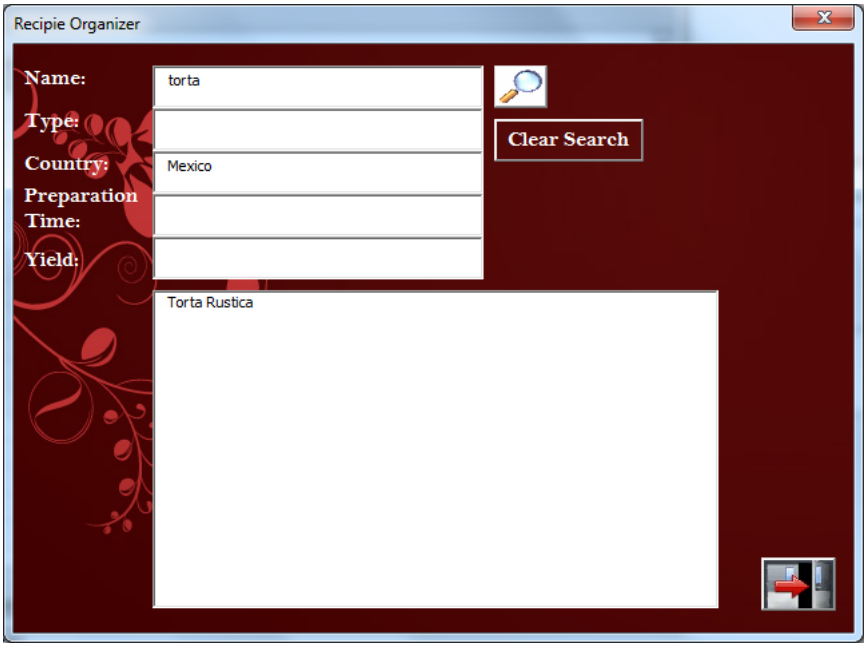
Advanced search: If the basic search is not enough, the user can do two types of advanced search on the local spreadsheet.

The user can select from two advanced search options by clicking on the radio buttons. Bas on the user's selection a new window will appear to allow the user to specify the search criteria

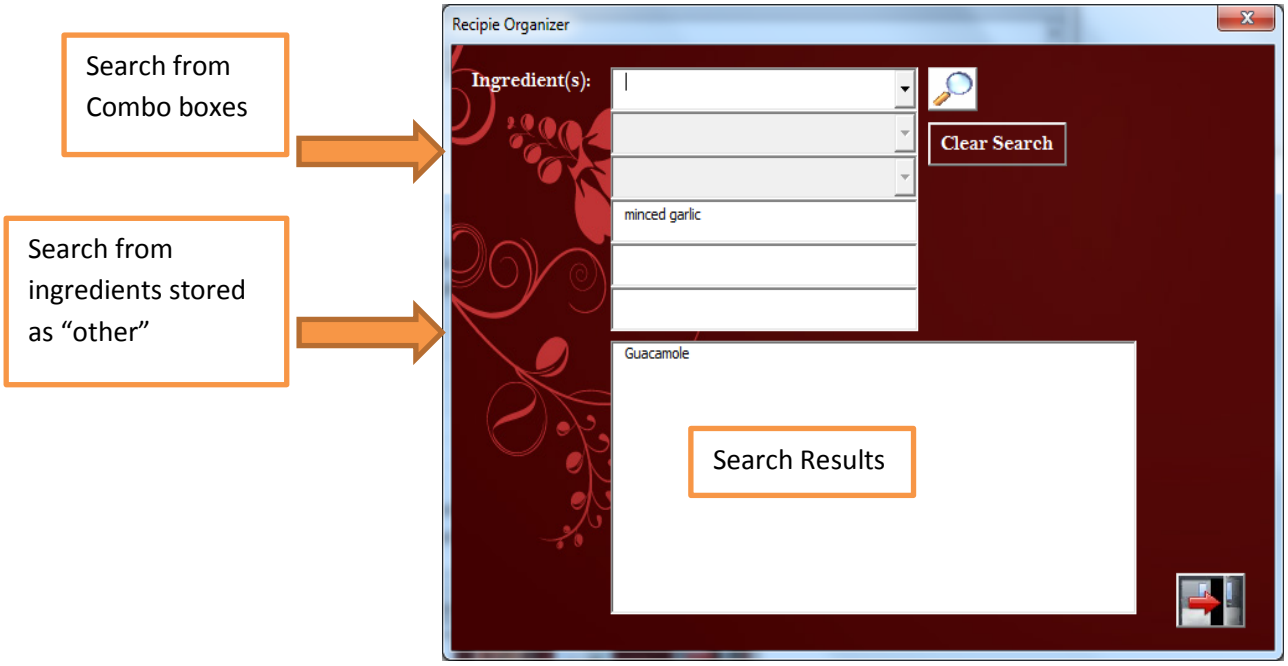
By name/type: this search is more than just name and type. In fact, this search allows the user to type up to five search criteria (name, type, country, preparation time, and yield). The search engine will display the results on a list and from there the user will have the choice to view/edit the recipe.

Search criteria specified by the user

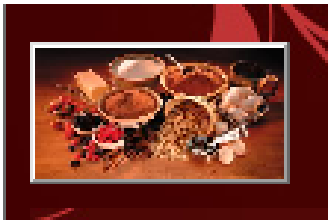
Results from Search



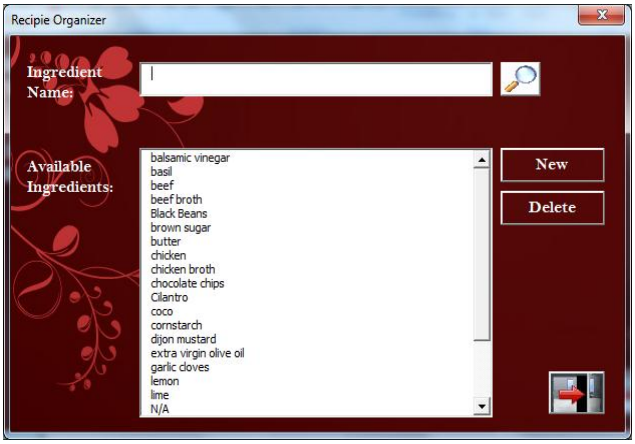
By ingredients: if the user selects this option, there will be two kinds of inputs to specify search criteria. There are 3 combo boxes that allow any combination of 1-3 ingredients. These ingredients are the ones that populate the forms through the application. However, the application provides some extra flexibility for the user that wants to add a onetime ingredient and ingredient information to a recipe without having to save this data as a default ingredient. This is defined as "other" in the ingredient criteria. This part of the search allows the user to specify the "other" information that needs to be selected to find results. Like in the previous example, the results will be displayed in the list.



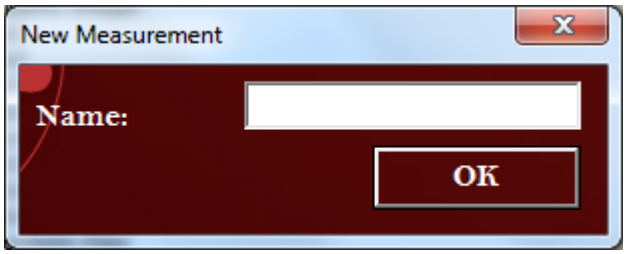
Add/Edit/Delete ingredients: the user can click on the following buttons to add/delete the ingredients that populate the ingredient combo boxes through the application.



The left button allows the user to display the form on the right and search and select ingredients. Once an ingredient is selected, the delete button will remove it from the list



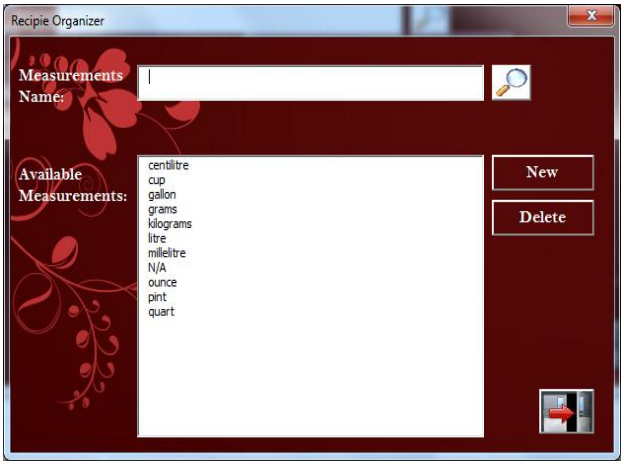
If the user wants to create a new ingredient the “new” button must be clicked and the following form will appear, in which the user can specify the name of the new ingredient that will be available in the application:



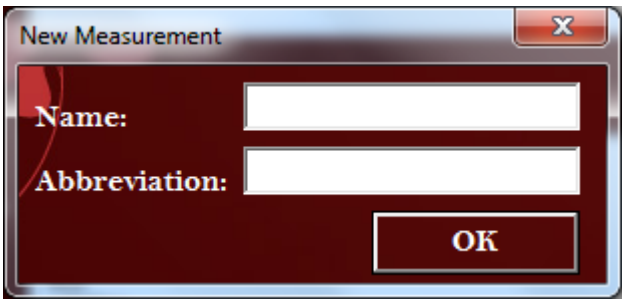
Once the user clicks on the “ok” button the ingredient will be added to the excel spreadsheet.

Add/Edit/Delete measurements: the user can click on the following button to add/delete the measurements that populate the measurement combo boxes through the application.

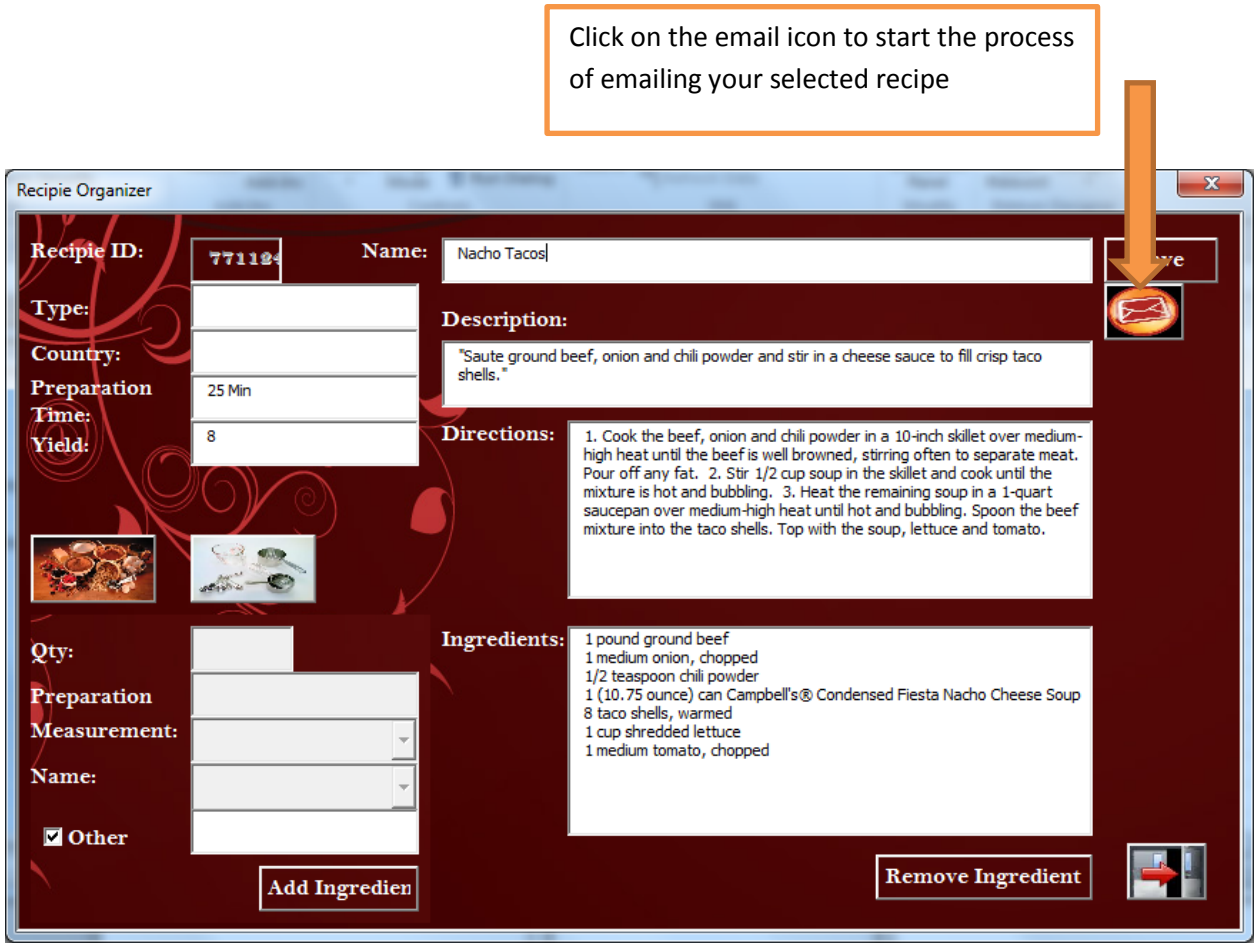
The left button allows the user to display the form on the right and search and select measurements. Once a measurement is selected the delete button will remove it from the list

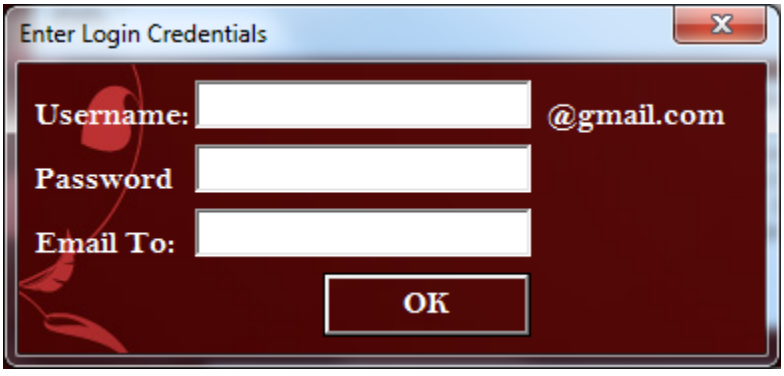


If the user wants to create a new measurement, the “new” button must be clicked and a new window will appear where the user can specify the measurement as well as the abbreviation for it. Once the new measurement is saved, the user will have access to it in the entire application through the measurements combo boxes.



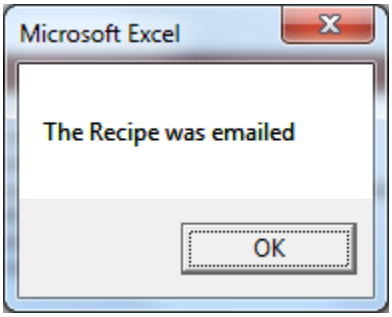
Email recipe: Once the user has selected an open the recipe that needs to be emailed, the user will have to click on the email icon to call a new form where email credentials as well as destination emails must be entered.



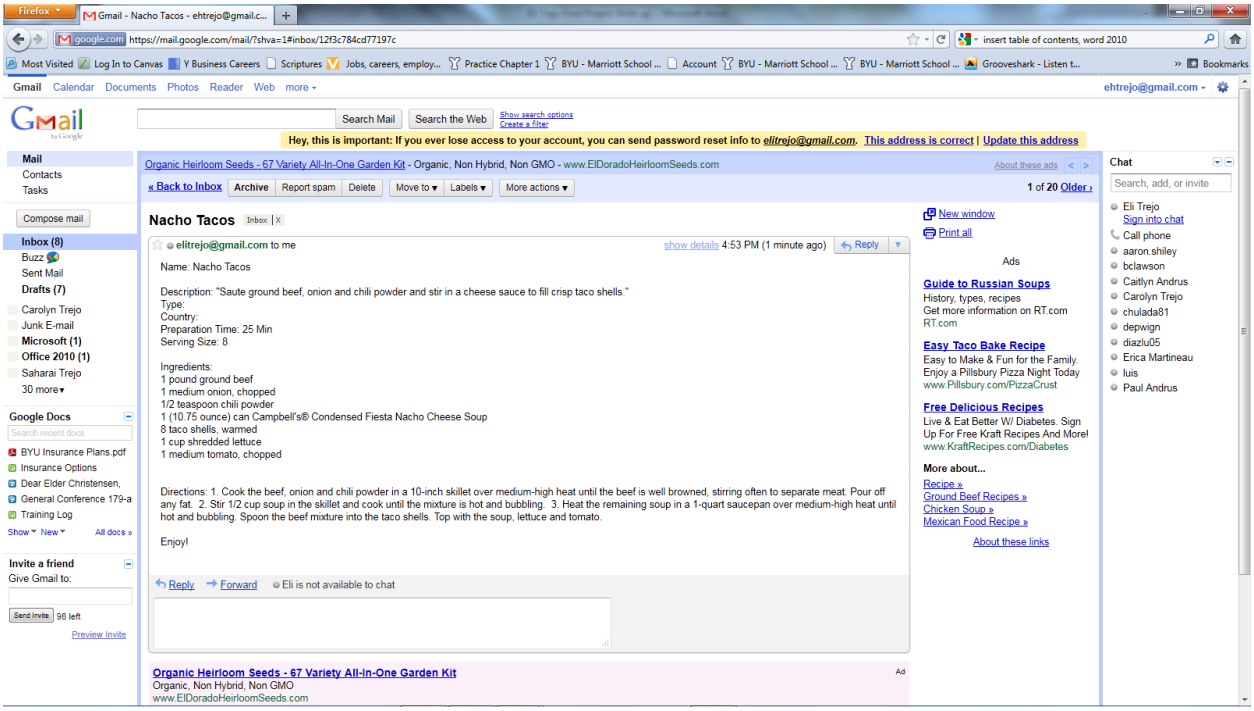


The email application only works for gmail accounts. The use has to specify his/her email credentials and at least one destination email address. When the user clicks the OK button the application emails the selected recipe.

If the recipe is successfully emailed the user is immediately notified in a message box.



The recipient of the recipe immediately gets an email with the title of the recipe that was sent as well as with an organized copy of the entire recipe.



Learning and conceptual difficulties

For all the previous assignments I was able to know exactly what tools I needed to use and it was fairly simple to map the solution before creating the implementation. I feel that with my current project I failed to do more detail planning before implementation. I thought I had spent enough hours planning the architecture of my application. I felt that with some diagrams and notes I had enough direction to start building my recipe organizer. Nevertheless, as soon as the construction of my application started, I made a few changes to my design and it didn't take too long before I was overwhelmed by the assignment.

As I started to code it became apparent that my design was changing. Small and major changes made me do things two and in some cases three times. Although I value that time as learning, I feel that I could have been a bit more efficient if I had taking more time planning before executing.

One of the exciting parts about my application is the way in which the data is stored. As mentioned before, the application stored data in such a way that it makes it simple to export the data to a database without violating the integrity of the data and without having to make changes to database tables. However, this means that through the entire application I had to read letters and write numbers and I had to read numbers and display letters to the user. This way of reading and writing data is seamless to the user but it caused me to write a lot more code than what I was expecting because the data to do data matching was spread in several spreadsheets rather than in one. I learned that first hand that best programming practices means efficiency for the user and initially a bit more work for the programmer.

What I liked about the project is that it gave me the opportunity to apply every concept learned in class. From customizing the ribbon, to getting data from the web, to reading files, manipulating data across multiple spread sheets, etc.; indeed, the project was a good capstone to practice what I learned during the semester.

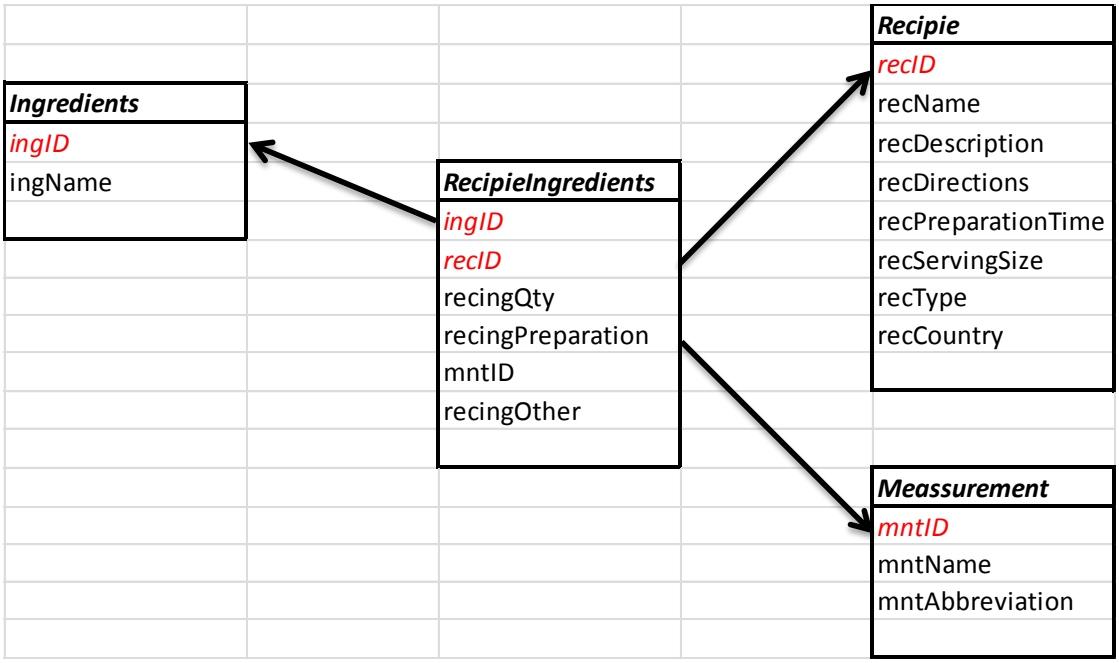
Also, during the project I had to find most of my answers on my own. I guess I became quite good at finding the answers on Google and knowing how to customize them to my application. However, this process took too much time.

The major challenge that I had in my application was coding part of the advanced search. Although I am providing everything that I committed to do upon submission of my proposal, my search is not quite as good as I wanted it to be. It returns the results that the user specifies but in the search by ingredient limits the user to search either by ingredients present in combo boxes or by random ingredients that the user can specify in a different data set. Ideally, I should be able to combine both searches. I feel that my code is pretty much ready to do the search that I want to do but there are a few bugs. For this reason, I am somewhat limiting the search capabilities of the search by ingredient section. I want to mention that I spend several hours on this section and I requested help. However, the solution might be a project by itself. What I learned is that the fact that I am providing more flexibility to the user to allow him to save

time and store data according to his/her convenience is somewhat limiting the capabilities of my applications or complicating my life as a programmer.

Write up detail

Data storage design: I think it is worth to mention that one of the first steps in my project was the design of data storage. The following diagram describes the tables along with its primary keys and data relations of my project.



Start the application: I personally found that customizing the ribbon by modifying the xml file was a bit easier, for me, than doing it thought the excel downloaded from the web. I want the user (my wife) to have a simple button in the home tab to start the forms. In the XML file I was able to indicate the tab position and to put the image of my choice (not form MS) as an ICON. When I made the changes to the XML file I also had to create a new procedure in one the modules to connect by customize ribbon to my initial application form.

When the user clicks the "Recipe Organizer" Icon from the Ribbon the startRecipeOrganizer procedure is called which in turn loads the main form (frmRecipieMain). The misspelling errors in the programming code are constant throughout the application so I decided to leave them like that and no try to correct them.

When the frmRecipieMain is loaded, three procedures are called immediately to sort the recipes, the ingredients, and the ingredients in recipes stored in different sheets. Each procedure is stored in a global

module because they are used in different forms. I programed my searched by recording macros and copying the code generated when the data is selected and sorted in a customized way.

Once the data is sorted I ask excel to read all the existing recipes and add them to the main list of my frmRecipeMain. My code makes sure that no recipe is added twice and that the list is cleared and uploaded every time that the main form is loaded.

Add a new recipe manually: When the user click on the NEW button, on the main form, the frmRecipieNew is loaded and shown. At initialize time, the form frmRecipieNew calls a procedure that generates 6 random numbers and prepopulates the txtRecId to display the number. Although the number might seem useless to the user, the application will grab this random number to store it in the recipe once the recipe is created.

With the exception of the recipe ID, which is automatically generated, the user is only required to type the recipe name. When the users clicks on the save button and if statement checks that the recipe name is provided. If a recipe name exists then the program finds the last row where the new recipe can be stored in the recipe sheet. Once the row is identified, the program takes all the input from the boxes and stores the data on the recipe Sheet.

If the data is successfully saved, the new recipe ID, as well as its row number, is stored in global variable for use in other forms. The code unloads the form and immediately loads form frmRecipie which among other things, it allows users to add ingredients to the recipe.

Upon loading form frmRecipie retrieves the stored recipe ID and row number for the location of the recipe on the spreadsheet. Based on that information it populates the text boxes and it loads ingredients and measurement information. Also, the form used the the “findIngredients” procedure to load all the ingredients already existing in the selected recipe. This procedure goes to the recipeIngredients sheet and based on the recipe selected in finds all the ingredients stored in the recipe. Once it finds the ingredients it does some matching and concatenation to display information from multiple boxes in a single line for the user to read. The concatenated information is displayed in the list fox found on frmRecipie.

Whenever the user adds or removes ingredients from the recipe the changes are automatically saved. However, when changes to any other part of the recipe are made the user has to click on the save button to make sure the changes are saved. Upon clicking the save button the program takes the recipe ID from the recipe and deletes all the basic information and replaces it with the data displayed in the text boxes. Once the changes are made the program notifies with a message box to the user that the transaction was successful.

If the user decides to edit the ingredients from a recipe he/she just has to click on the ingredient, shown in the ingredients list, that needs to be modified. A new window will appear with the selected

ingredient. The program takes the ingredient ID as well as the recipe ID to identify the ingredient in the sRecipeIngredients sheets. Once the item is identified the ingredient is deleted and replaced with the information that the user typed in the new window. After the changes are completed the program unloads the ingredients list in the frmRecipe and loads it with the new information.

Add a new recipe from the web: On the form frmRecipeMain the user has to type the name of the recipe that needs to be added to the recipe organizer application. Then, the user needs to select the "Web" radio button and click on the search ICON. Upon clicking the search icon the program will make sure that a name was typed in the text box provided for recipe names. If proper data is available the program checks that the web radio button is selected and if selected the function findInAllRecipies will be called. This function calls Internet explorer and calls for www.allrecipes.com to be open. The name of the recipe that was typed by the user is passed in the URL and the list of the search results are displayed in the browser.

When the user selects one of the search results do display the details of a recipe the program considers the data displayed on the browser as ready to be imported. The button IMPORT becomes enable and when the user clicks on it the program reads all the data from the browser and creates a new sheet on the workbook. The data from the browser is copied into the new excel sheet (which is named with the name of the user's search). Once the data is on the sheet a procedure looks for key words and relative positioning to see what data is available and write it as a new recipe in different parts of the workbook. When the data is copied, the program deletes the form that was recently created.

The program includes codes such as ON ERROR GO TO to make sure that if part of the data that the program is trying to import to the spreadsheet is not available the program can keep running and importing the data that is there. As soon as the new recipe is imported the user can see the title of it in the recipe list on the mail form of the program.

Edit an existing recipe: On the main form of the program (frmRecipeMain) the user can see all the available recipes in the program. By double clicking on the recipe list, the program will read the recipe ID and it will identify the row number of the location of the recipe in the sRecipie sheet. Since the recipes are sorted basic math calculation allows the program to identify the row location. Once these two variables are identified and stored, the program loads the frmRecipie form which has all the information of the recipe.

As explained in previous sections, when the frmRecipie is loaded it used the row number and the recipe ID of the recipe to pull out the data from the sheets and display the information on the form. Any changes that the user makes to the main form can be saved with the SAVE button. When this button is clicked the program deletes all the information of the recipe (except the recipe ID) and replaces is it with the data in the text boxes.

The adding, removing and editing of existing ingredients belonging to a recipe are done in the ingredients section. The section “Add a new recipe manually” explains in detail how this changes are made by the program. Please refer to that section for more detail.

Delete an existing recipe: in the main form of the program, frmRecipieMain, a complete list of the available recipes is displayed. To delete a specific recipe the user needs to select it in the list. If the recipe is selected it will be highlighted. Then, the user can click on the delete button to remove the recipe form the recipe organizer.

The program makes sure that the user selects a recipe before executing the action of deletion with an if statement that reads the list index of the item selected. By identified the item that was selected the program can determine the row number of the spreadsheet where the main body of the recipe is stored. Once the row number is identified the program finds the recipe ID number by referring to the location of the columns of the sRecipie sheet. Upon identification of the recipe ID the program goes to the srecipieIngredients sheet and identifies all the ingredients that belong to the selected recipe by looping through each of them. When an ingredient is identified, the entire row is deleted. As soon as all the ingredients belonging to the recipe selected by the user are removed from the recipe (the ingredient itself is not deleted from the program. Only the relation of ingredient/Recipe is deleted) the program removes the recipe itself by deleting the entire row where the recipe is located on the sRecipie Sheet.

The user will notice that as soon as the recipe is removed it will no longer be visible in the list of available recipes. To update the list the program unloads the items in the list and then reads all the recipes available, after deletion of the selected recipe, and adds them to the list.

Search for an existing recipe: In the main form of the program, frmRecipieMain, the user has a complete list of the available recipes is displayed. The list is automatically sorted by alphabetical order and the user can easily scroll down to find the desired recipe. Also the user can type the name of a recipe on the recipe name text box and select the list radio button (selected by default). Then, the search button can be clicked. As soon as this button is clicked the program will match the name of the recipe with the InStr function to see if that text is part of the title of an existing recipe. if a match is found the program will retrieve the recipe ID of the recipe found an use it to highlight the recipe in the list by using the list index property.

If the user desires to do an advance search two options are available. 1) The search by name and type, and 2) search by ingredients. Both advanced search options are available on the main form frmRecipieMain and can be activated by clicking on the radio button of the desired search.

By Name/Type: The search by name/type option loads the frmRecipeSearchNT, which upon loading provides a list of text boxes where the user can identify the name, type, country, preparation time, and yield of a recipe. To do the search the user has to provide at least one search criteria. The search icon can be clicked to get the search results, which will be displayed in the results list box. When the user clicks on the search icon the program reads the content on the text boxes and determines how many criteria will need to be met to identify possible results. A local variable that counts the search criteria the user provided is increased every time that valid text is found on each text box provided to the user. Since all the information for this part of the search is located in one sheet, the program loops through each of the available recipes in the sRecipie sheet and checks that all the criteria is met comparing strings in the text boxes with the text stored in the spreadsheet. Each time that a recipe meets all the search criteria, it is added to an array that will be used to pull our results and populate the list results once the program has looped and compared the details of each recipe.

By ingredient: If the user decides to do an advanced search by ingredients then he/she needs to double click on that radio button. This will load the frmSearchI form which provides 3 combo boxes populated with all the ingredients available in the program, as well as text boxes for the user to specify ingredients that were saves as OTHER.

The user can select up to 3 ingredients from the combo box. The second and third combo box becomes available only when the user has selected an item in the previous combo box. If the user selects items from the combo boxes and the search button can be clicked. This will allow the program to read the selection of ingredients (previously to this the program makes sure that different ingredients are selected by checking the list index of the combo boxes).

The program determines how many items ingredients the user specified. Once this information is known the program creates an array of all the recipes that have at least the same number of ingredients that the results must have. Then, it loops trough each of this recipes in the sRecipieIngredients sheet to see if the recipe has all the ingredients specified by the user. When a recipe meets the criteria, the program adds the recipe ID to a new array of results. This array is used to display results on the results list.

If the user decides to type the ingredients to find them on the other column, the recipe follows a similar process to the one described in the previous paragraph. The major difference is that compares the text on the OTHER column of the sRecipieIngredients sheets with the text that the user provided. When a match is found the recipe ID is also added to the results array.

Clear search: these options are available in both types of search. The code of this button clear the text boxes, unselect the combo boxes, and clears the list results of each search.

Viewing search results: on both types of search the user can double click an item in the list results, which are storing the recipe ID. When the user double clicks on the results the frmRecipieMain is called and all the pertaining data to the selected recipe is display.

Add/Edit/Delete set ingredients: on the main recipe form the user can click on the ingredients ICON which will load the frmRecipieIngredients form. The form will display information in a similar layout to the rest of the program. There is a search box at the top where the user can easily navigate to the desired ingredient. Also, the user can directly select the ingredient in the list box. Once the ingredient is selected the delete button can be clicked to delete the ingredient.

The delete button gets the ingredient id and it first verifies that no recipes are using the ingredient. There are some functions that search for the ingredient in the ingredients/recipe sheets and if the ingredients are not found then the button delete the entire row of the location of the ingredient in the ingredients sheet.

To add a new ingredient the user needs to click on the new button which will call anew form where the user will be able to specify the name of the new ingredient. The form will automatically generate an ingredient ID by using a function stored in the general modules. Then it will find the last row in the ingredients sheet and go to the next row to insert the generated ID as well as the name of the ingredient. When the ingredient is added the user will be able to see it in the ingredients list.

Add/Edit/Delete set measurements: on the main recipe form the user can click on the ingredients ICON which will load the frmRecipieMeassurements form. The form will display information in a similar layout to the rest of the program. There is a search box at the top where the user can easily navigate to the desired measurement. Also, the user can directly select the measurement in the list box. Once the measurement is selected the delete button can be clicked to delete the measurement.

The delete button gets the measurement id and it first verifies that no recipes are using the measurement. There are some functions that search for the measurement in the ingredients/recipe sheets and if the ingredients are not found then the button delete the entire row of the location of the measurement in the measurement sheet.

To add a new measurement the user needs to click on the new button which will call anew form where the user will be able to specify the name and abbreviation of the new measurement. The form will automatically generate a measurement ID by using a function stored in the general modules. Then it will find the last row in the ingredients sheet and go to the next row to insert the generated ID as well as the name of the ingredient. When the ingredient is added the user will be able to see it in the ingredients list.

Email recipe: Once the user has selected a recipe and the recipe is displayed in the frmRecipe form, the user can click the email ICON to start the process of emailing the recipe. This Icon calls the “sendMessage" procedure which creates a template variable that refers to a text file titled “Recipe.txt”. This file has the basic template with tags of the message that will be email to a user.

A variable called message (data type string) stores the message of the mail and it takes its data from the text boxes of the recipe being display. For the ingredients sections in has to go thorugh a loop to make sure that all the ingredients are included in the same section.

Once the message is build, the frmPassword form is called. This form makes sure that the user provides an email account (only valid for gmail), a password (visually protected), and a destination address or addresses. Once the credentials are provided the sendGMail procedure is called. This procedure is stored in a global module and is the procedure used during class. If the message is sent the user is notified with a message from a message box.

Appendix

Application Code

frmAbrMeassurement

```
Option Explicit

Private Sub cmdOk_Click()
   If Trim(txtName) = "" Or Trim(txtAbbreviation) = "" Then
      MsgBox "Enter Name and Abbreviation"
   Else
      frmAbrMeassurement.Hide
   End If
End Sub
```

frmNewIngredient

```
Option Explicit

Private Sub cmdOk_Click()
   If Trim(txtName) = "" Then
      MsgBox "Enter the Name"
   Else
      frmNewIngredient.Hide
   End If
End Sub
```

frmPassword

```
Private Sub CommandButton1_Click()
 frmPassword.Hide
End Sub
```

frmRecipie

```
Option Explicit
Dim rowNumber As Long
Public TRN As Long
Public selRecID As Long
Public ingid As Long
Public other As Boolean

Private Sub cmdAddIngredient_Click()
   Dim iID As Long
   Dim mId As Long
   Dim ingredientSelected As Long
   Dim meassurementSelected As Long

   'check if other is selected
   If chkOther.Value = True Then
      If Trim(txtOther.text) = "" Then
         MsgBox "Enter Ingredient description before adding the ingredient"
         Exit Sub
      Else
         iID = get6RandomNumbers
         'before wriging make sure that the new recipe and ingredient ID' do not exist on the list
         If ingredientInRecipe(iID, txtRecId) = True Then
            MsgBox "The ingredient already exist on the Recipe"
```

```
            Exit Sub
        End If

        If Trim(sRecipieIngredients.Cells(2, 1)) = "" Then
            sRecipieIngredients.Cells(2, 1) = iID
            sRecipieIngredients.Cells(2, 2) = txtRecId.text
            sRecipieIngredients.Cells(2, 6) = txtOther.text
        Else
            sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row + 1), 1) = iID
            sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 2) = txtRecId.text
            sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 6) = txtOther.text
        End If
    End If
Else
    ingredientSelected = getCmdIngredientID
    meassurementSelected = getCmdMeassurementID

    If ingredientSelected = -1 Then
        iID = "123456"
    Else
        iID = sIngredients.Cells(ingredientSelected + 2, 1)
    End If

    If meassurementSelected = -1 Then
        mId = "123456"
    Else
        mId = sMeassurement.Cells(meassurementSelected + 2, 1)
    End If

    If (Trim(iID) = "" Or Trim(iID) = "123456") Or Trim(Trim(txtQty)) = "" Then
        MsgBox "Enter Ingredient and quantity specifications before saving"
        Exit Sub
    End If

    'before wriging make sure that the new recipe and ingredient ID' do not exist on the list
    If ingredientInRecipe(iID, txtRecId) = True Then
        MsgBox "The ingredient already exist on the Recipe"
        Exit Sub
    End If

    If Trim(sRecipieIngredients.Cells(2, 1)) = "" Then
        sRecipieIngredients.Cells(2, 1) = iID
        sRecipieIngredients.Cells(2, 2) = txtRecId
        sRecipieIngredients.Cells(2, 3) = Trim(txtQty)
        sRecipieIngredients.Cells(2, 4) = Trim(txtPreparation)
        sRecipieIngredients.Cells(2, 5) = mId
    Else
        sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row + 1), 1) = iID
        sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 2) = txtRecId
        sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 3) = Trim(txtQty)
        sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 4) = Trim(txtPreparation)
        sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 5) = mId
    End If


End If
```

```
    clearTxtBoxes
    refreshForm

End Sub

'Private Sub cmdClose_Click()
'    Unload Me
'    Unload frmRecipieMain
'    frmRecipieMain.Show
'
''    Unload frmRecipieMain
''    Unload frmRecipie
''    frmRecipie.Hide
'
''    frmRecipieMain.Hide
'
''    Load frmRecipieMain
''        frmRecipieMain.Show
'
'End Sub

'Private Sub cmdEmail_Click()
'    sendMessage
'End Sub

Sub sendMessage()
    Dim row As Integer
    Dim template As String
    Dim message As String
    Dim wasSent As Boolean
    Dim ingredients As String
    frmPassword.Show

    template = readFile(ThisWorkbook.path & "\Recipe.txt")

    row = 0
      message = Replace(template, "<Name>", txtRecName)
      message = Replace(message, "<Description>", txtRecDescription)
      message = Replace(message, "<Type>", txtRecType)
      message = Replace(message, "<Country>", txtRecCountry)
      message = Replace(message, "<Preparation Time>", txtPreparationTime)
      message = Replace(message, "<Serving Size>", txtServingSize)
      message = Replace(message, "<Directions>", txtRecDirections)

      For row = 0 To lstRecipieIngredients.ListCount - 1
        ingredients = ingredients & lstRecipieIngredients.List(row) & vbNewLine
      Next

      message = Replace(message, "<Ingredients>", ingredients)

      wasSent = sendGMail(frmPassword.txtEmailTo, frmPassword.txtUsername & "@gmail.com", frmPassword.txtPassword, _
                          txtRecName, message)

    If wasSent Then MsgBox "The Recipe was emailed"
'    sendGMail "8017920444@txt.att.net", frmPassword.txtUsername, frmPassword.txtPassword, _
'                          "", "Master, I am finished and await your orders."
End Sub
```

```
Public Sub refreshForm()
   Do While lstRecipieIngredients.ListCount > 0
    lstRecipieIngredients.RemoveItem 0
   Loop

   frmRecipieMain.recid = selRecID
   frmRecipieMain.rowNumber = rowNumber
   UserForm_Initialize
End Sub

Private Sub cmdRemoveIngredient_Click()
   Dim ingredientLocation As Long

   If lstRecipieIngredients.ListCount = 0 Or lstRecipieIngredients.listIndex < 0 Then
      MsgBox "An existing Ingredient must be selected before trying to delete it"
      Exit Sub
   End If

   'get the value of the location of the selected ingredient
   ingredientLocation = lstRecipieIngredients.listIndex + 1
   'get list number. Remember that the Ingredient is going to be equal to the list Index +1
   ingid = getIngredientID(ingredientLocation, selRecID)
   'find the row that matches the ingredientID and the recipie ID and delete the entire row
   deleteIngredient ingid, selRecID

   refreshForm

End Sub

Private Sub cmdSave_Click()
  Dim sRN As Long
   'get the recipe ID
   sRN = getRNwithRID(selRecID)

   'clear the cells for the selected recipe ID
   sRecipie.Range("B" & sRN & ":" & "H" & sRN).Clear

   'popullate those cells with the new information
   sRecipie.Cells(sRN, 2) = txtRecName
   sRecipie.Cells(sRN, 3) = txtRecDescription
   sRecipie.Cells(sRN, 4) = txtRecDirections
   sRecipie.Cells(sRN, 5) = txtPreparationTime
   sRecipie.Cells(sRN, 6) = txtServingSize
   sRecipie.Cells(sRN, 7) = txtRecType
   sRecipie.Cells(sRN, 8) = txtRecCountry

   'tell the user that the information was saved
   MsgBox "Your changes have been saved"
   refreshForm
End Sub



Private Sub imgEmail_Click()
   sendMessage
End Sub
```

```
Private Sub imgExit_Click()
    Unload Me
    Unload frmRecipieMain
    frmRecipieMain.Show
End Sub

Private Sub imgIngredients_Click()
    frmRecipieIngredients.Show
End Sub

Private Sub imgMeasurements_Click()
    frmRecipieMeassurements.Show
End Sub

Private Sub lstRecipieIngredients_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
    Dim ingredientLocation As Long
    Dim otherValues As String

    If lstRecipieIngredients.ListCount = 0 Or lstRecipieIngredients.listIndex < 0 Then
        MsgBox "An existing Ingredient must be selected before trying to edit it"
        Exit Sub
    End If

    ingredientLocation = lstRecipieIngredients.listIndex + 1
    'get list number. Remember that the Ingredient is going to be equal to the list Index +1
    ingid = getIngredientID(ingredientLocation, selRecID)
    otherValues = getOtherValue(ingid, selRecID)

    If Len(Trim(otherValues)) > 0 Then
        other = True
    Else
        other = False
    End If
    frmRecipieIngredientsEdit.Show
End Sub

Sub reloadForm()
    UserForm_Initialize
End Sub
Private Sub UserForm_Initialize()
'    sortRecipes
'    sortIngredients
'    sortMeassurements
    other = False

    selRecID = frmRecipieMain.recid
    'the first time the form is shown make sure that is readable but now editable
    'lockTextBoxes

    'get row number of selected recipie
    rowNumber = frmRecipieMain.rowNumber
    TRN = rowNumber
    'popullate text boxes
    popullateTextBoxes
    loadMeasurements
    loadIngredients
```

```
    lockInputBoxes

    'find all the ingredients related to the recipie and add them to the list
    findIngredients sRecipie.Cells(frmRecipieMain.rowNumber, 1).Value
End Sub

'Private Sub lockTextBoxes()
'
'    txtPreparationTime.Enabled = False
'    txtPreparationTime.BackColor = "&H80000004"
'
'    txtRecCountry.Enabled = False
'    txtRecCountry.BackColor = "&H80000004"
'
'    txtRecDescription.Enabled = False
'    txtRecDescription.BackColor = "&H80000004"
'
'    txtRecName.Enabled = False
'    txtRecName.BackColor = "&H80000004"
'
'    txtRecType.Enabled = False
'    txtRecType.BackColor = "&H80000004"
'
'    txtServingSize.Enabled = False
'    txtServingSize.BackColor = "&H80000004"
'
'    txtRecDirections.Enabled = False
'    txtRecDirections.BackColor = "&H80000004"
'
''    lstRecipieIngredients.Enabled = False
''    lstRecipieIngredients.BackColor = "&H80000004"
''
''    cmdAddIngredient.Enabled = False
''    cmdRemoveIngredient.Enabled = False
'End Sub
'
'Private Sub unlockTextBoxes()
'
'    txtPreparationTime.Enabled = True
'    txtPreparationTime.BackColor = "&H80000005"
'
'    txtRecCountry.Enabled = True
'    txtRecCountry.BackColor = "&H80000005"
'
'    txtRecDescription.Enabled = True
'    txtRecDescription.BackColor = "&H80000005"
'
'    txtRecName.Enabled = True
'    txtRecName.BackColor = "&H80000005"
'
'    txtRecType.Enabled = True
'    txtRecType.BackColor = "&H80000005"
'
'    txtServingSize.Enabled = True
'    txtServingSize.BackColor = "&H80000005"
'
'    txtRecDirections.Enabled = True
```

```
'    txtRecDirections.BackColor = "&H80000005"
'
''    lstRecipieIngredients.Enabled = True
''    lstRecipieIngredients.BackColor = "&H80000005"
''
''    cmdAddIngredient.Enabled = True
''    cmdRemoveIngredient.Enabled = True
'End Sub


Sub popullateTextBoxes()
    txtRecId = sRecipie.Cells(rowNumber, 1)
    txtRecName = sRecipie.Cells(rowNumber, 2)
    txtRecDescription = sRecipie.Cells(rowNumber, 3)
    txtRecDirections = sRecipie.Cells(rowNumber, 4)
    txtPreparationTime = sRecipie.Cells(rowNumber, 5)
    txtServingSize = sRecipie.Cells(rowNumber, 6)
    txtRecType = sRecipie.Cells(rowNumber, 7)
    txtRecCountry = sRecipie.Cells(rowNumber, 8)
End Sub

Sub findIngredients(recipiID As Long)
    Dim startingRow As Long
    Dim lstIndex As Long
    Dim condensedIngredientItem As String
    Dim ingredientQuantity As String
    Dim ingredientMeasurement As String
    Dim ingredienPreparation As String
    Dim ingredientName As String
    Dim ingredientOther As String

    startingRow = 2
    'get all the ingredients that match the recipi ID
    Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 2))) > 0
        If Trim(sRecipieIngredients.Cells(startingRow, 2)) = recipiID Then
            'get Ingredient Quantity
            ingredientQuantity = getIngredientQuantity(sRecipieIngredients.Cells(startingRow, 1), sRecipieIngredients.Cells(startingRow, 2))

            'get Measurement
            ingredientMeasurement = getMeasurementID(sRecipieIngredients.Cells(startingRow, 1), sRecipieIngredients.Cells(startingRow, 2))
            If LCase(Trim(ingredientMeasurement)) = "n/a" Then ingredientMeasurement = ""

            'get Ingrediet Preparation
            ingredienPreparation = getIngredientPreparation(sRecipieIngredients.Cells(startingRow, 1), sRecipieIngredients.Cells(startingRow, 2))

            'get Ingredient Name
            ingredientName = getIngredientName(sRecipieIngredients.Cells(startingRow, 1))
            If LCase(Trim(ingredientName)) = "n/a" Then ingredientName = ""

            If Len(Trim(sRecipieIngredients.Cells(startingRow, 6))) < 1 Then
                condensedIngredientItem = ingredientQuantity & " " & ingredientMeasurement & " " & ingredienPreparation & " " & ingredientName
            Else
                condensedIngredientItem = sRecipieIngredients.Cells(startingRow, 6)
            End If

            'add items to list
            lstRecipieIngredients.AddItem condensedIngredientItem
```

```
      End If
      startingRow = startingRow + 1
   Loop

End Sub

'get Ingredient Quantity
Function getIngredientQuantity(ingredientID As Long, recipieID As Long)
   Dim startingRow As Long
   startingRow = 2

   getIngredientQuantity = ""

   Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
      If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
         getIngredientQuantity = sRecipieIngredients.Cells(startingRow, 3)
         Exit Function
      End If
      startingRow = startingRow + 1
   Loop
End Function

'get Measurement
Function getMeasurementID(ingredientID As Long, recipieID As Long)
   Dim startingRow As Long
   Dim measurementId As Long

   startingRow = 2

   getMeasurementID = -1

   Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
      If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
         measurementId = sRecipieIngredients.Cells(startingRow, 5)
         getMeasurementID = measurementIDtoAbbreviation(measurementId)
         Exit Function
      End If
      startingRow = startingRow + 1
   Loop
End Function

Function measurementIDtoAbbreviation(measurementId As Long)
   Dim startingRow As Long
   startingRow = 2

   Do While Len(Trim(sMeassurement.Cells(startingRow, 1))) > 0
      If sMeassurement.Cells(startingRow, 1) = measurementId Then
         measurementIDtoAbbreviation = sMeassurement.Cells(startingRow, 3)
         Exit Function
      End If
      startingRow = startingRow + 1
   Loop
End Function

'get Ingrediet Preparation
Function getIngredientPreparation(ingredientID As Long, recipieID As Long)
   Dim startingRow As Long
```

```
    startingRow = 2

    getIngredientPreparation = ""

    Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
        If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
            getIngredientPreparation = sRecipieIngredients.Cells(startingRow, 4)
            Exit Function
        End If
        startingRow = startingRow + 1
    Loop
End Function


'get Ingredient Name
Function getIngredientName(ingredientID As Long)
    Dim startingRow As Long
    startingRow = 2
    getIngredientName = ""
    Do While Len(Trim(sIngredients.Cells(startingRow, 1))) > 0
        If sIngredients.Cells(startingRow, 1) = ingredientID Then
            getIngredientName = sIngredients.Cells(startingRow, 2)
            Exit Function
        End If
        startingRow = startingRow + 1
    Loop
End Function

Function getIngredientID(ingredientLocation As Long, recipieID As Long)
    Dim maxIterations As Long
    Dim startingRow As Long

    startingRow = 2

    Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
        If sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
            maxIterations = maxIterations + 1
        End If

        If maxIterations = ingredientLocation Then
            getIngredientID = sRecipieIngredients.Cells(startingRow, 1)
            Exit Do
        End If

        startingRow = startingRow + 1
    Loop

End Function

Function getOtherValue(ingredientID As Long, recipieID As Long)
    Dim maxIterations As Long
    Dim startingRow As Long

    startingRow = 2
    getOtherValue = ""

    Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
```

```
    If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
      getOtherValue = sRecipieIngredients.Cells(startingRow, 6)
      Exit Function
    End If
    startingRow = startingRow + 1
  Loop
End Function

Sub deleteIngredient(ingredientID As Long, recipieID As Long)
  Dim maxIterations As Long
  Dim startingRow As Long

  startingRow = 2

  Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
    If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
      sRecipieIngredients.Cells(startingRow, 6).EntireRow.Delete
      Exit Sub
    End If
    startingRow = startingRow + 1
  Loop
End Sub

Function getRNwithRID(recid As Long)
  Dim rN As Long

  rN = 2
  Do While Len(Trim(sRecipie.Cells(rN, 1))) > 0
    If sRecipie.Cells(rN, 1) = recid Then
      getRNwithRID = rN
      Exit Do
    End If
    rN = rN + 1
  Loop
End Function

'Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
'   cmdClose_Click
'End Sub

Private Sub chkOther_Click()
  If chkOther.Value = True Then
    'lock the rest of the input boxes
    lockInputBoxes
  Else
    'unlock the rest of the input boxes
    unlockInputBoxes
  End If
End Sub

Sub lockInputBoxes()
  chkOther = True
  txtOther.Enabled = True
  txtOther.BackColor = "&H80000005"

  txtQty.Enabled = False
  txtQty.BackColor = "&H80000004"
```

```
    txtPreparation.Enabled = False
    txtPreparation.BackColor = "&H80000004"

    cmbMeasurements.Enabled = False
    cmbMeasurements.BackColor = "&H80000004"

    txtPreparation.BackColor = "&H80000004"

    cboIngredients.Enabled = False
    cboIngredients.BackColor = "&H80000004"

    txtPreparation.BackColor = "&H80000004"
End Sub

Sub unlockInputBoxes()
    chkOther = False
    txtOther.Enabled = False
    txtOther.BackColor = "&H80000004"

    txtQty.Enabled = True
    txtQty.BackColor = "&H80000005"

    txtPreparation.Enabled = True
    txtPreparation.BackColor = "&H80000005"

    cmbMeasurements.Enabled = True
    cmbMeasurements.BackColor = "&H80000005"
    txtPreparation.BackColor = "&H80000005"

    cboIngredients.Enabled = True
    cboIngredients.BackColor = "&H80000005"

    txtPreparation.BackColor = "&H80000005"
End Sub

'get Ingredient selected from combo box
Function getCmdIngredientID() As Long
    getCmdIngredientID = cboIngredients.listIndex
End Function

'get Measurement selected from combo box
Function getCmdMeassurementID() As Long
    getCmdMeassurementID = cmbMeasurements.listIndex
End Function

Sub loadMeasurements()
    Dim rowNum As Long
    Dim sellItem As Long
    rowNum = 2

    Do While Len(Trim(sMeassurement.Cells(rowNum, 1))) > 0
        cmbMeasurements.AddItem sMeassurement.Cells(rowNum, 2)
        rowNum = rowNum + 1
    Loop
End Sub
```

```
'load ingredients
Sub loadIngredients()
   Dim rowNum As Long
   Dim selItem As Long
   rowNum = 2

   Do While Len(Trim(sIngredients.Cells(rowNum, 1))) > 0
      cboIngredients.AddItem sIngredients.Cells(rowNum, 2)
      rowNum = rowNum + 1
   Loop
End Sub

Sub clearTxtBoxes()
   txtOther = ""
   txtQty = ""
   txtPreparation.Enabled = True
   cmbMeasurements.listIndex = -1
   cboIngredients.listIndex = -1
End Sub
```

frmRecipieIngredients

```
Option Explicit

Public rowNumber As Long
Public recipieFoundID As Long
Public recid As Long
Dim agent1 As New agent
Dim recipieListIndex As Long

'Dim foundByName As Boolean

'Private Sub cmdClose_Click()
'    Unload frmRecipieIngredients
'End Sub

Private Sub cmdDelete_Click()
   Dim rowToDelete As Long
   Dim test As String

   If lstIngredients.ListCount = 0 Or lstIngredients.listIndex < 0 Then
      MsgBox "An existing Ingredient must be selected before trying to delete it"
      Exit Sub
   End If

   rowToDelete = lstIngredients.listIndex + 2 ' 4
   test = sIngredients.Cells(rowToDelete, 1) '2
   If ingredientInUse(test) = False Then
      sIngredients.Cells(rowToDelete, 1).EntireRow.Delete
      GoTo 1
   Else
      MsgBox "A Recipe is using this ingredient. Delete the recipe before deleting the ingredient", vbCritical
      Exit Sub
   End If

   Do While lstIngredients.ListCount > 0
    lstIngredients.RemoveItem 0
   Loop
```

```
1:
   sortIngredients

   Do While lstIngredients.ListCount > 0
      lstIngredients.RemoveItem 0
   Loop

   rowNumber = 2
   Do While Len(Trim(sIngredients.Cells(rowNumber, 2))) > 0
     lstIngredients.AddItem (Trim(sIngredients.Cells(rowNumber, 2)))
     rowNumber = rowNumber + 1
   Loop
   rowNumber = 2
End Sub

Private Sub cmdNew_Click()
   Dim rowN As Long
   Dim newIngredient As String

   frmNewIngredient.Show
   newIngredient = frmNewIngredient.txtName
   If Trim(newIngredient) <> "" Then
      If ingredientExist(newIngredient) = False Then
         If Trim(sIngredients.Cells(2, 2)) = "" Then
            rowN = 2
         Else
            rowN = sIngredients.Cells(1, 1).End(xlDown) + 1
         End If
            sIngredients.Cells(rowN, 1) = get6RandomNumbers
            sIngredients.Cells(rowN, 2) = Trim(newIngredient)
      Else
         MsgBox "The ingredient already exists"
         Exit Sub
      End If
   End If
   Unload frmNewIngredient
   sortIngredients

   Do While lstIngredients.ListCount > 0
      lstIngredients.RemoveItem 0
   Loop

   rowNumber = 2
   Do While Len(Trim(sIngredients.Cells(rowNumber, 2))) > 0
     lstIngredients.AddItem (Trim(sIngredients.Cells(rowNumber, 2)))
     rowNumber = rowNumber + 1
   Loop
   rowNumber = 2
End Sub

Private Sub cmdSearch_Click()
   'check for a valid input criteria
   If Trim(txtRecName.Value) = "" Then
      MsgBox "Enter the name of an ingredient before doing a search", vbCritical, "Missing Recipie Name"
   Else
      recipieFoundID = searchByName(Trim(txtRecName.Value))
```

```
      'if record is found make sure to highlight the record
      If recipieFoundID = 0 Then
         MsgBox "The Ingredient is not in the list", vbInformation, "Recipie not found"
      Else
         lstIngredients.listIndex = rowNumber - 2
      End If
   End If
End Sub


Private Sub imgExit_Click()
   Unload frmRecipieIngredients
End Sub

Private Sub imgSearch_Click()
   cmdSearch_Click
End Sub

Private Sub UserForm_Initialize()
   Dim selectedtRange As Range

   rowNumber = 2

   'sort the list of recipies you want to display
   sortIngredients

   'get a list of all the recipie titles
   Do While lstIngredients.ListCount > 0
    lstIngredients.RemoveItem 0
   Loop

   Do While Len(Trim(sIngredients.Cells(rowNumber, 2))) > 0
      lstIngredients.AddItem (Trim(sIngredients.Cells(rowNumber, 2)))
      rowNumber = rowNumber + 1
   Loop
End Sub

Private Function searchByName(ingredientName As String) As Long
   rowNumber = 2
   recipieListIndex = 0

   'get a list of all the recipie titles
   Do While Len(Trim(sIngredients.Cells(rowNumber, 2))) > 0

      If InStr(1, LCase(Trim(sIngredients.Cells(rowNumber, 2))), LCase(Trim(ingredientName))) > 0 Then
         'foundByName = True
         searchByName = sIngredients.Cells(rowNumber, 1)
         Exit Do
      End If
      rowNumber = rowNumber + 1
      recipieListIndex = recipieListIndex + 1
   Loop
End Function


Private Function ingredientExist(ingName As String) As Boolean
   ingredientExist = False
```

```
    On Error GoTo 1
    sIngredients.Select
    Columns("B:B").Select
    Selection.find(What:=ingName, After:=ActiveCell, LookIn:=xlFormulas, _
        LookAt:=xlWhole, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
        MatchCase:=False, SearchFormat:=False).Activate
        ingredientExist = True
        GoTo 2
1:
    ingredientExist = False
    Exit Function
2:

End Function

Private Function ingredientInUse(ingid As String) As Boolean
    ingredientInUse = False
    On Error GoTo 1
    sRecipieIngredients.Select
    Columns("A:A").Select
    Selection.find(What:=ingid, After:=ActiveCell, LookIn:=xlFormulas, _
        LookAt:=xlWhole, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
        MatchCase:=False, SearchFormat:=False).Activate
        ingredientInUse = True
        GoTo 2
1:
    ingredientInUse = False
    Exit Function
2:

End Function
```

frmRecipieIngredientsEdit

```
Option Explicit

Public RID As Long
Public iID As Long

Private Sub chkOther_Click()
    If chkOther.Value = True Then
        'lock the rest of the input boxes
        lockInputBoxes
    Else
        'unlock the rest of the input boxes
        unlockInputBoxes
    End If
End Sub

Private Sub cmdSave_Click()
    Dim begRow As Long
    Dim ingredientSelected As Long
    Dim meassurementSelected As Long
    Dim mId As Long
    Dim qty As String
    Dim preparation As String

    begRow = 2
```

```
'go to the recipie Ingredients sheet and delete the row for the rID and iID
Do While Len(Trim(sRecipieIngredients.Cells(begRow, 1))) > 0
   If sRecipieIngredients.Cells(begRow, 1) = iID And sRecipieIngredients.Cells(begRow, 1) = iID Then
      sRecipieIngredients.Cells(begRow, 1).EntireRow.Delete
      Exit Do
   End If
   begRow = begRow + 1
Loop

If chkOther.Value = True Then
   If Trim(txtOther.Value) = "" Then
      MsgBox "Enter Ingredient description before saving data"
      Exit Sub
   Else
      iID = get6RandomNumbers
      'before wriging make sure that the new recipe and ingredient ID' do not exist on the list
      If ingredientInRecipe(iID, RID) = True Then
         MsgBox "The ingredient already exist on the Recipie"
         Exit Sub
      End If

      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row + 1), 1) = iID
      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 2) = RID
      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 6) = txtOther.text

   End If
Else
'in the recipie Ingredients sheet write the new values specified in the form
   'read all inputs
   ingredientSelected = getCmdIngredientID
   meassurementSelected = getCmdMeassurementID

   If ingredientSelected = -1 Then
      iID = "123456"
   Else
      iID = sIngredients.Cells(ingredientSelected + 2, 1)
   End If

   If meassurementSelected = -1 Then
      mId = "123456"
   Else
      mId = sMeassurement.Cells(meassurementSelected + 2, 1)
   End If

   'before wriging make sure that the new recipe and ingredient ID' do not exist on the list
   If ingredientInRecipe(iID, RID) = True Then
      MsgBox "The ingredient already exist on the Recipie"
      Exit Sub
   End If

   qty = txtQty
   preparation = txtPreparation

   sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row + 1), 1) = iID
   sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 2) = RID
   sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 3) = qty
```

```
        sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 4) = preparation
        sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 5) = mId

    End If
    Unload Me
    frmRecipie.refreshForm

End Sub

Private Sub UserForm_Initialize()
    RID = frmRecipie.selRecID
    iID = frmRecipie.ingid

    'sort the ingredients
    sortIngredients
    sortMeassurements

    'load meassurements and ingredients
    loadMeasurements
    loadIngredients

    'based on the ingredient ID and Recipie ID find all relevant information and display values
    txtQty = frmRecipie.getIngredientQuantity(iID, RID)
    txtPreparation = frmRecipie.getIngredientPreparation(iID, RID)

    'select meassurements
    selectMeassurement

    'select ingredients
    selectIngredient

    chkOther = frmRecipie.other

    If chkOther.Value = True Then
        chkOther = True
        lockInputBoxes

        txtOther.visible = True
        Dim rN As Double
        rN = 2
        Do While Len(Trim(sRecipieIngredients.Cells(rN, 1))) > 0
            If Trim(sRecipieIngredients.Cells(rN, 1)) = iID And Trim(sRecipieIngredients.Cells(rN, 2)) = RID Then
                txtOther.text = Trim(sRecipieIngredients.Cells(rN, 6))
                Exit Do
            End If
            rN = rN + 1
        Loop
    Else
        chkOther = False
        unlockInputBoxes
    End If
End Sub

Sub loadMeasurements()
    Dim rowNum As Long
    Dim selItem As Long
    rowNum = 2
```

```
    Do While Len(Trim(sMeassurement.Cells(rowNum, 1))) > 0
        cmbMeasurements.AddItem sMeassurement.Cells(rowNum, 2)
        rowNum = rowNum + 1
    Loop

End Sub

Sub selectMeassurement()
    Dim rowNum As Long
    Dim sellItem As Long
    rowNum = 2

    sellItem = getMeasurementID(iID, RID)

    If Len(Trim(sellItem)) > 0 Then
        Do While Len(Trim(sMeassurement.Cells(rowNum, 1))) > 0
            If Trim(sMeassurement.Cells(rowNum, 1)) = Trim(sellItem) Then
                cmbMeasurements.listIndex = rowNum - 2
                Exit Do
            End If
            rowNum = rowNum + 1
        Loop
    Else
        cmbMeasurements.listIndex = -1
    End If
End Sub


Function getMeasurementID(ingredientID As Long, recipieID As Long) As Long
    Dim startingRow As Long

    startingRow = 2

    Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
        If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
            getMeasurementID = sRecipieIngredients.Cells(startingRow, 5)
            Exit Function
        End If
        startingRow = startingRow + 1
    Loop
End Function


'load ingredients
Sub loadIngredients()
    Dim rowNum As Long
    Dim sellItem As Long
    rowNum = 2

    Do While Len(Trim(sIngredients.Cells(rowNum, 1))) > 0
        cboIngredients.AddItem sIngredients.Cells(rowNum, 2)
        rowNum = rowNum + 1
    Loop
End Sub

Sub selectIngredient()
```

```
    Dim rowNum As Long
    Dim selItem As Long
    rowNum = 2

    selItem = iID

    If Len(Trim(selItem)) > 0 Then
        Do While Len(Trim(sIngredients.Cells(rowNum, 1))) > 0
            If Trim(sIngredients.Cells(rowNum, 1)) = Trim(selItem) Then
                cboIngredients.listIndex = rowNum - 2
                Exit Do
            End If
            rowNum = rowNum + 1
        Loop
    Else
        cboIngredients.listIndex = -1
    End If
End Sub


Sub lockInputBoxes()
    txtOther.Enabled = True
    txtOther.BackColor = "&H80000005"

    txtQty.Enabled = False
    txtQty.BackColor = "&H80000004"

    txtPreparation.Enabled = False
    txtPreparation.BackColor = "&H80000004"

    cmbMeasurements.Enabled = False
    cmbMeasurements.BackColor = "&H80000004"

    txtPreparation.BackColor = "&H80000004"

    cboIngredients.Enabled = False
    cboIngredients.BackColor = "&H80000004"

    txtPreparation.BackColor = "&H80000004"
End Sub

Sub unlockInputBoxes()
    txtOther.Enabled = False
    txtOther.BackColor = "&H80000004"

    txtQty.Enabled = True
    txtQty.BackColor = "&H80000005"

    txtPreparation.Enabled = True
    txtPreparation.BackColor = "&H80000005"

    cmbMeasurements.Enabled = True
    cmbMeasurements.BackColor = "&H80000005"
    txtPreparation.BackColor = "&H80000005"

    cboIngredients.Enabled = True
    cboIngredients.BackColor = "&H80000005"
```

```
    txtPreparation.BackColor = "&H80000005"
End Sub

'get Ingredient selected from combo box
Function getCmdIngredientID() As Long
    getCmdIngredientID = cboIngredients.listIndex
End Function

'get Measurement selected from combo box
Function getCmdMeassurementID() As Long
    getCmdMeassurementID = cmbMeasurements.listIndex
End Function
```

frmRecipieMain

```
Option Explicit

Public rowNumber As Long
Public recipieFoundID As Long
Public recid As Long
Dim agent1 As New agent
Dim recipieListIndex As Long

'Dim foundByName As Boolean

'Private Sub cmdClose_Click()
'    'unload and exit frmRecipieMain
'    frmRecipieMain.Hide
'    Unload frmRecipieMain
'End Sub

Private Sub cmdDelete_Click()
    Dim rowToDelete As Long
    Dim test As Long

    If lstRecipies.ListCount = 0 Or lstRecipies.listIndex < 0 Then
        MsgBox "An existing Recipe must be selected before trying to delete it"
        Exit Sub
    End If

    rowToDelete = lstRecipies.listIndex + 2 ' 4
    test = sRecipie.Cells(rowToDelete, 1) '2

    If rowToDelete >= 2 Then
        deleteRecipie sRecipie.Cells(rowToDelete, 1), rowToDelete
    Else
        MsgBox "SYSTEM ERROR: Sorry for the incovenience, but the application can't delete the Recipe that you selected.", vbCritical
    End If


    Do While lstRecipies.ListCount > 0
     lstRecipies.RemoveItem 0
    Loop

    rowNumber = 2
    'get a list of all the recipie titlesv
    Do While Len(Trim(sRecipie.Cells(rowNumber, 2))) > 0
```

```
      lstRecipies.AddItem (Trim(sRecipie.Cells(rowNumber, 2)))
      rowNumber = rowNumber + 1
   Loop
   rowNumber = 2

End Sub

Private Sub cmdImport_Click()
  Dim recRN As Long
  Dim i As Long
  Dim ingRN As Long
  Dim activeRN As Long
  Dim rName As String
  Dim preparationTime As String
  Dim ingredients As String
  Dim newRID As Long
  Dim recDirections As String
  Dim placeToSearch As Range

  If Trim(sRecipie.Cells(2, 1)) = "" Then
    recRN = 2
  Else
    recRN = sRecipie.Cells(1, 1).End(xlDown).row + 1
  End If

  If Trim(txtRecName) = "" And optCmdWeb = False Then
    MsgBox "Before importing, 1)type a recipie name, 2)Select the WEB option, 3)Click the SEARCH button, and 4)Make sure that your IE displays
the recipie that you want to import", vbCritical
    Exit Sub
  End If
  rName = Trim(txtRecName)
  'create a new sheet and import all the data from the web
  agent1.importPage rName

  newRID = get6RandomNumbers



  'generate Recipe ID
  sRecipie.Cells(recRN, 1) = newRID

  On Error GoTo 1

  'Preparation time
    Sheets(rName).Select
    Sheets(rName).Range("A1").Select
    Sheets(rName).Cells.find(What:="Ready In:", After:=ActiveCell, LookIn:=xlFormulas, _
    LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
    MatchCase:=False, SearchFormat:=False).Activate
    ActiveCell.Offset(1).Activate
    'write Preparation time
    sRecipie.Cells(recRN, 5) = ActiveCell

    GoTo 2
1:
    Sheets(rName).Cells.find(What:="Ingredients", After:=ActiveCell, LookIn:=xlFormulas, _
    LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
```

```
    MatchCase:=False, SearchFormat:=False).Activate
    Cells.FindNext(After:=ActiveCell).Activate
    ActiveCell.Offset(2).Activate

    If Trim(sRecipieIngredients.Cells(2, 1)) = "" Then
      ingRN = 2
    Else
      ingRN = sRecipieIngredients.Cells(1, 1).End(xlDown).row + 1
    End If

    Do While Len(Trim(ActiveCell)) > 0
       'write ingredient ID
       sRecipieIngredients.Cells(ingRN, 1) = get6RandomNumbers
       'write recipe ID
       sRecipieIngredients.Cells(ingRN, 2) = newRID
       'write ingrediet
       sRecipieIngredients.Cells(ingRN, 6) = ActiveCell

       ActiveCell.Offset(1).Activate
       ingRN = ingRN + 1
    Loop

    GoTo 3
2:
On Error GoTo 3
    Sheets(rName).Cells.find(What:="Ingredients", After:=ActiveCell, LookIn:=xlFormulas, _
    LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
    MatchCase:=False, SearchFormat:=False).Activate
    ActiveCell.Offset(2).Activate

    ingRN = sRecipieIngredients.Cells(1, 1).End(xlDown).row + 1

    Do While Len(Trim(ActiveCell)) > 0
       'write ingredient ID
       sRecipieIngredients.Cells(ingRN, 1) = get6RandomNumbers
       'write recipe ID
       sRecipieIngredients.Cells(ingRN, 2) = newRID
       'write ingrediet
       sRecipieIngredients.Cells(ingRN, 6) = ActiveCell

       ActiveCell.Offset(1).Activate
       ingRN = ingRN + 1
    Loop

3:
On Error GoTo 4
    Sheets(rName).Cells.find(What:="Directions", After:=ActiveCell, LookIn:=xlFormulas, _
    LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
    MatchCase:=False, SearchFormat:=False).Activate
     ActiveCell.Offset(2).Activate

    Do While Len(Trim(ActiveCell)) > 0
       recDirections = recDirections & ActiveCell & " "
       ActiveCell.Offset(1).Activate
    Loop

    'write directions
```

```
    sRecipie.Cells(recRN, 4) = recDirections

4:
On Error GoTo 5

    Sheets(rName).Select
    Sheets(rName).Cells.find(What:="Nutritional Information", After:=ActiveCell, LookIn:= _
        xlFormulas, LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:= _
        xlNext, MatchCase:=False, SearchFormat:=False).Activate
    Cells.FindNext(After:=ActiveCell).Activate
    ActiveCell.Offset(1).Activate

    'write Recipe Title
    sRecipie.Cells(recRN, 2) = Trim(ActiveCell)
5:
On Error GoTo 6
    Sheets(rName).Select
    Sheets(rName).Range("A1").Select
    Sheets(rName).Cells.find(What:="by", After:=ActiveCell, LookIn:=xlFormulas, _
    LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
    MatchCase:=False, SearchFormat:=False).Activate
    ActiveCell.Offset(1).Activate

    'write description
    sRecipie.Cells(recRN, 3) = ActiveCell.Value

6:
On Error GoTo 7
    Sheets(rName).Select
    Sheets(rName).Range("A1").Select
    Sheets(rName).Cells.find(What:="Servings Per Recipe", After:=ActiveCell, LookIn:=xlFormulas, _
    LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
    MatchCase:=False, SearchFormat:=False).Activate

    Dim sPs As String

    sPs = ActiveCell
    sPs = Replace(sPs, "Servings Per Recipe: ", "")
    sRecipie.Cells(recRN, 6) = Trim(sPs)

7:
    Application.DisplayAlerts = False
    Sheets(rName).Select
    Sheets(rName).Delete
    Application.DisplayAlerts = True

    Unload Me
    frmRecipieMain.Show


'   MsgBox "The application experienced some errors but probably part of your recipe was still imported"

End Sub
```

```
Private Sub cmdNew_Click()
   frmRecipieNew.Show
End Sub

Private Sub imgExit_Click()
   'unload and exit frmRecipieMain
   frmRecipieMain.Hide
   Unload frmRecipieMain
End Sub

'Private Sub cmdSearch_Click()
'
'    'check for a valid input criteria
'    If Trim(txtRecName.Value) = "" Then
'        MsgBox "please enter a valid Recipie Name before doing a search", vbCritical, "Missing Recipie Name"
'    Else
'        If optCmdList = True Then
'            'search specified name in current list
'            recipieFoundID = searchByName(Trim(txtRecName.Value))
'            'if record is found make sure to highlight the record
'            If recipieFoundID = 0 Then
'                MsgBox "The Recipie is not in the list", vbInformation, "Recipie not found"
'            Else
'
'                lstRecipies.listIndex = rowNumber - 2
'            End If
'        Else
'            'Search web
'            findInAllRecipies Trim(txtRecName.Value)
'            cmdImport.Enabled = True
'        End If
'    End If
'End Sub

Private Sub imgIngredients_Click()
   frmRecipieIngredients.Show
End Sub

Private Sub imgMeasurements_Click()
   frmRecipieMeassurements.Show
End Sub

Private Sub imgSearch_Click()

   'check for a valid input criteria
   If Trim(txtRecName.Value) = "" Then
      MsgBox "please enter a valid Recipie Name before doing a search", vbCritical, "Missing Recipie Name"
   Else
      If optCmdList = True Then
         'search specified name in current list
         recipieFoundID = searchByName(Trim(txtRecName.Value))
         'if record is found make sure to highlight the record
         If recipieFoundID = 0 Then
            MsgBox "The Recipie is not in the list", vbInformation, "Recipie not found"
         Else

            lstRecipies.listIndex = rowNumber - 2
```

```
        End If
      Else
        'Search web
        findInAllRecipies Trim(txtRecName.Value)
        cmdImport.Enabled = True
      End If
    End If

End Sub

Private Sub lstRecipies_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
    If lstRecipies.listIndex + 2 < 2 Then
      Exit Sub
    End If

    rowNumber = lstRecipies.listIndex + 2
    recid = sRecipie.Cells(rowNumber, 1)
    Load frmRecipie
'   frmRecipie.Show
    UserForm_Initialize
    frmRecipie.Show
End Sub

Private Sub optIngredients_Click()
    frmRecipieSearchI.Show
End Sub

Private Sub optNameType_Click()
    frmRecipieSearchNT.Show
End Sub

Private Sub UserForm_Initialize()
    Dim selectedtRange As Range

    rowNumber = 2

    'sort the list of recipies you want to display
    sortRecipes
    sortIngredients
    sortMeassurements

    'automatically set search list as the defaul searh
    optCmdList = True

    'get a list of all the recipie titles
    Do While lstRecipies.ListCount > 0
     lstRecipies.RemoveItem 0
    Loop

    Do While Len(Trim(sRecipie.Cells(rowNumber, 2))) > 0
      lstRecipies.AddItem (Trim(sRecipie.Cells(rowNumber, 2)))
      rowNumber = rowNumber + 1
    Loop
End Sub

Private Function searchByName(recipieName As String) As Long
    rowNumber = 2
```

```
    recipieListIndex = 0

    'get a list of all the recipie titles
    Do While Len(Trim(sRecipie.Cells(rowNumber, 2))) > 0

        If InStr(1, LCase(Trim(sRecipie.Cells(rowNumber, 2))), LCase(Trim(recipieName))) > 0 Then
            'foundByName = True
            searchByName = sRecipie.Cells(rowNumber, 1)
            Exit Do
        End If
        rowNumber = rowNumber + 1
        recipieListIndex = recipieListIndex + 1
    Loop
End Function

Sub findInAllRecipies(name As String)
    agent1.visible = True
    AppActivate "explorer"
    agent1.openpage "http://allrecipes.com/Search/Recipes.aspx?WithTerm=" & URLEncode(name)
    agent1.savePage
End Sub
```

frmRecipieMeassurements

```
Option Explicit

Public rowNumber As Long
Public recipieFoundID As Long
Public recid As Long
Dim agent1 As New agent
Dim recipieListIndex As Long

'Dim foundByName As Boolean

'Private Sub cmdClose_Click()
'    Unload frmRecipieMeassurements
'End Sub

Private Sub cmdDelete_Click()
    Dim rowToDelete As Long
    Dim test As String

    If lstMeassurements.ListCount = 0 Or lstMeassurements.listIndex < 0 Then
        MsgBox "An existing Meassurement must be selected before trying to delete it"
        Exit Sub
    End If

    rowToDelete = lstMeassurements.listIndex + 2 ' 4
    test = sMeassurement.Cells(rowToDelete, 1) '2
    If meassurementInUse(test) = False Then
        sMeassurement.Cells(rowToDelete, 1).EntireRow.Delete
        GoTo 1
    Else
        MsgBox "A Recipe is using this Meassurement. Delete the recipe before deleting the meassurement", vbCritical
        Exit Sub
    End If

    Do While lstMeassurements.ListCount > 0
```

```
    lstMeassurements.RemoveItem 0
  Loop

1:
  sortMeassurements

  Do While lstMeassurements.ListCount > 0
    lstMeassurements.RemoveItem 0
  Loop

  rowNumber = 2
  Do While Len(Trim(sMeassurement.Cells(rowNumber, 2))) > 0
    lstMeassurements.AddItem (Trim(sMeassurement.Cells(rowNumber, 2)))
    rowNumber = rowNumber + 1
  Loop
  rowNumber = 2
End Sub

Private Sub cmdNew_Click()
  Dim rowN As Long
  Dim newMeassurement As String
  Dim newAbbreviation As String

  frmAbrMeassurement.Show

  If Trim(frmAbrMeassurement.txtName) = "" Or Trim(frmAbrMeassurement.txtAbbreviation) = "" Then
    MsgBox "Make sure to enter the name and abbreviation"
    Exit Sub
  Else
    newMeassurement = frmAbrMeassurement.txtName
    newAbbreviation = frmAbrMeassurement.txtAbbreviation


    If meassurementExists(newMeassurement) = False Then
      If Trim(sMeassurement.Cells(2, 2)) = "" Then
        rowN = 2
      Else
        rowN = sMeassurement.Cells(1, 1).End(xlDown) + 1
      End If
        sMeassurement.Cells(rowN, 1) = get6RandomNumbers
        sMeassurement.Cells(rowN, 2) = Trim(newMeassurement)
        sMeassurement.Cells(rowN, 3) = Trim(newAbbreviation)
        Unload frmAbrMeassurement
    Else
      MsgBox "The Measurement already exists"
      Exit Sub
    End If
  End If


  sortMeassurements

  Do While lstMeassurements.ListCount > 0
    lstMeassurements.RemoveItem 0
  Loop

  rowNumber = 2
```

```
  Do While Len(Trim(sMeassurement.Cells(rowNumber, 2))) > 0
    lstMeassurements.AddItem (Trim(sMeassurement.Cells(rowNumber, 2)))
    rowNumber = rowNumber + 1
  Loop
  rowNumber = 2
End Sub

Private Sub cmdSearch_Click()
  'check for a valid input criteria
  If Trim(txtRecName.Value) = "" Then
     MsgBox "Enter the name of an ingredient before doing a search", vbCritical, "Missing Recipie Name"
  Else
     recipieFoundID = searchByName(Trim(txtRecName.Value))
     'if record is found make sure to highlight the record
     If recipieFoundID = 0 Then
        MsgBox "The Ingredient is not in the list", vbInformation, "Recipie not found"
     Else
        lstMeassurements.listIndex = rowNumber - 2
     End If
  End If
End Sub


Private Sub imgExit_Click()
  Unload Me ' frmRecipieIngredients
End Sub


Private Sub imgSearch_Click()
  cmdSearch_Click
End Sub

Private Sub UserForm_Initialize()
  Dim selectedtRange As Range

  rowNumber = 2

  'sort the list of recipies you want to display
  sortMeassurements

  'get a list of all the recipie titles
  Do While lstMeassurements.ListCount > 0
   lstMeassurements.RemoveItem 0
  Loop

  Do While Len(Trim(sMeassurement.Cells(rowNumber, 2))) > 0
     lstMeassurements.AddItem (Trim(sMeassurement.Cells(rowNumber, 2)))
     rowNumber = rowNumber + 1
  Loop
End Sub

Private Function searchByName(ingredientName As String) As Long
  rowNumber = 2
  recipieListIndex = 0

  'get a list of all the recipie titles
  Do While Len(Trim(sMeassurement.Cells(rowNumber, 2))) > 0
```

```
      If InStr(1, LCase(Trim(sMeassurement.Cells(rowNumber, 2))), LCase(Trim(ingredientName))) > 0 Then
        'foundByName = True
        searchByName = sMeassurement.Cells(rowNumber, 1)
        Exit Do
      End If
      rowNumber = rowNumber + 1
      recipieListIndex = recipieListIndex + 1
    Loop
End Function


Private Function meassurementExists(meassurementName As String) As Boolean
    meassurementExists = False
    On Error GoTo 1
    sMeassurement.Select
    Columns("B:B").Select
    Selection.find(What:=meassurementName, After:=ActiveCell, LookIn:=xlFormulas, _
      LookAt:=xlWhole, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
      MatchCase:=False, SearchFormat:=False).Activate
      meassurementExists = True
      GoTo 2
1:
    meassurementExists = False
    Exit Function
2:

End Function

Private Function meassurementInUse(measurementId As String) As Boolean
    meassurementInUse = False
    On Error GoTo 1
    sRecipieIngredients.Select
    Columns("E:E").Select
    Selection.find(What:=measurementId, After:=ActiveCell, LookIn:=xlFormulas, _
      LookAt:=xlWhole, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
      MatchCase:=False, SearchFormat:=False).Activate
      meassurementInUse = True
      GoTo 2
1:
    meassurementInUse = False
    Exit Function
2:

End Function
```

frmRecipieNew

```
Option Explicit

'Private Sub cmdClose_Click()
'    Unload Me
'End Sub


Private Sub cmdSave_Click()
    Dim sRN As Long
```

```
    If Trim(txtRecName) = "" Then
        MsgBox "Please specify at least the Recipe Name before saving", vbCritical
        Exit Sub
    End If

    If Trim(sRecipie.Cells(2, 1)) = "" Then
        sRN = 2
    Else
        sRN = sRecipie.Cells(1, 1).End(xlDown).row + 1
    End If


    'popullate those cells with the new information
    sRecipie.Cells(sRN, 1) = txtRecId.Value
    sRecipie.Cells(sRN, 2) = txtRecName
    sRecipie.Cells(sRN, 3) = txtRecDescription
    sRecipie.Cells(sRN, 4) = txtRecDirections
    sRecipie.Cells(sRN, 5) = txtPreparationTime
    sRecipie.Cells(sRN, 6) = txtServingSize
    sRecipie.Cells(sRN, 7) = txtRecType
    sRecipie.Cells(sRN, 8) = txtRecDescription
    sRecipie.Cells(sRN, 4) = txtRecDirections
    sRecipie.Cells(sRN, 5) = txtPreparationTime
    sRecipie.Cells(sRN, 6) = txtServingSize
    sRecipie.Cells(sRN, 7) = txtRecType
    sRecipie.Cells(sRN, 8) = txtRecCountry


'    sortRecipes
'    sortIngredients
'    sortMeassurements



    frmRecipieMain.recid = txtRecId.Value
    frmRecipieMain.rowNumber = sRN

    Unload Me
    frmRecipie.Show

End Sub

Private Sub imgExit_Click()
    Unload Me
End Sub

Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
    'cmdClose_Click
    Unload Me
End Sub

Private Sub UserForm_Initialize()
    Dim sixRN As Long

    sixRN = get6RandomNumbers
    txtRecId.text = sixRN
End Sub
```

frmRecipieNTFound

```
Option Explicit
Dim rowNumber As Long
Public TRN As Long
Public selRecID As Long
Public ingid As Long
Public other As Boolean

Private Sub cmdAddIngredient_Click()
  Dim iID As Long
  Dim mId As Long
  Dim ingredientSelected As Long
  Dim meassurementSelected As Long

  'check if other is selected
  If chkOther.Value = True Then
    If Trim(txtOther.text) = "" Then
      MsgBox "Enter Ingredient description before adding the ingredient"
      Exit Sub
    Else
      iID = get6RandomNumbers
      'before wriging make sure that the new recipe and ingredient ID' do not exist on the list
      If ingredientInRecipe(iID, txtRecId) = True Then
        MsgBox "The ingredient already exist on the Recipe"
        Exit Sub
      End If

      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row + 1), 1) = iID
      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 2) = txtRecId.text
      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 6) = txtOther.text

    End If
  Else
    ingredientSelected = getCmdIngredientID
    meassurementSelected = getCmdMeassurementID

    If ingredientSelected = -1 Then
      iID = "123456"
    Else
      iID = sIngredients.Cells(ingredientSelected + 2, 1)
    End If

    If meassurementSelected = -1 Then
      mId = "123456"
    Else
      mId = sMeassurement.Cells(meassurementSelected + 2, 1)
    End If

    If (Trim(iID) = "" Or Trim(iID) = "123456") Or Trim(Trim(txtQty)) = "" Then
      MsgBox "Enter Ingredient and quantity specifications before saving"
      Exit Sub
    End If

    'before wriging make sure that the new recipe and ingredient ID' do not exist on the list
    If ingredientInRecipe(iID, txtRecId) = True Then
      MsgBox "The ingredient already exist on the Recipe"
      Exit Sub
```

```
      End If

      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row + 1), 1) = iID
      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 2) = txtRecId
      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 3) = Trim(txtQty)
      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 4) = Trim(txtPreparation)
      sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row), 5) = mId

   End If

   clearTxtBoxes
   refreshForm

End Sub

'Private Sub cmdClose_Click()
'    Unload Me
'
'
'
'End Sub

'Private Sub cmdEmail_Click()
'    sendMessage
'End Sub

Sub sendMessage()
   Dim row As Integer
   Dim template As String
   Dim message As String
   Dim wasSent As Boolean
   Dim ingredients As String
   frmPassword.Show

   template = readFile(ThisWorkbook.path & "\Recipe.txt")

   row = 0
     message = Replace(template, "<Name>", txtRecName)
     message = Replace(message, "<Description>", txtRecDescription)
     message = Replace(message, "<Type>", txtRecType)
     message = Replace(message, "<Country>", txtRecCountry)
     message = Replace(message, "<Preparation Time>", txtPreparationTime)
     message = Replace(message, "<Serving Size>", txtServingSize)
     message = Replace(message, "<Directions>", txtRecDirections)

     For row = 0 To lstRecipieIngredients.ListCount - 1
       ingredients = ingredients & lstRecipieIngredients.List(row) & vbNewLine
     Next

     message = Replace(message, "<Ingredients>", ingredients)

     wasSent = sendGMail(frmPassword.txtEmailTo, frmPassword.txtUsername & "@gmail.com", frmPassword.txtPassword, _
                         txtRecName, message)

   If wasSent Then MsgBox "The Recipe was emailed"
'    sendGMail "8017920444@txt.att.net", frmPassword.txtUsername, frmPassword.txtPassword, _
'                          "", "Master, I am finished and await your orders."
```

```
End Sub

Public Sub refreshForm()
   Do While lstRecipieIngredients.ListCount > 0
    lstRecipieIngredients.RemoveItem 0
   Loop

   frmRecipieMain.recid = selRecID
   frmRecipieMain.rowNumber = rowNumber
   UserForm_Initialize
End Sub

Private Sub cmdRemoveIngredient_Click()
   Dim ingredientLocation As Long

   If lstRecipieIngredients.ListCount = 0 Or lstRecipieIngredients.listIndex < 0 Then
      MsgBox "An existing Ingredient must be selected before trying to delete it"
      Exit Sub
   End If

   'get the value of the location of the selected ingredient
   ingredientLocation = lstRecipieIngredients.listIndex + 1
   'get list number. Remember that the Ingredient is going to be equal to the list Index +1
   ingid = getIngredientID(ingredientLocation, selRecID)
   'find the row that matches the ingredientID and the recipie ID and delete the entire row
   deleteIngredient ingid, selRecID

   refreshForm

End Sub

Private Sub cmdSave_Click()
  Dim sRN As Long
   'get the recipe ID
   sRN = getRNwithRID(selRecID)

   'clear the cells for the selected recipe ID
   sRecipie.Range("B" & sRN & ":" & "H" & sRN).Clear

   'popullate those cells with the new information
   sRecipie.Cells(sRN, 2) = txtRecName
   sRecipie.Cells(sRN, 3) = txtRecDescription
   sRecipie.Cells(sRN, 4) = txtRecDirections
   sRecipie.Cells(sRN, 5) = txtPreparationTime
   sRecipie.Cells(sRN, 6) = txtServingSize
   sRecipie.Cells(sRN, 7) = txtRecType
   sRecipie.Cells(sRN, 8) = txtRecCountry

   'tell the user that the information was saved
   MsgBox "Your changes have been saved"
   refreshForm
End Sub



Private Sub imgEmail_Click()
   sendMessage
```

```
End Sub

Private Sub imgExit_Click()
   Unload Me
End Sub

Private Sub lstRecipieIngredients_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
   MsgBox "you can only add or remove ingredients from this form"
'    Dim ingredientLocation As Long
'    Dim otherValues As String
'
'    If lstRecipieIngredients.ListCount = 0 Or lstRecipieIngredients.listIndex < 0 Then
'       MsgBox "An existing Ingredient must be selected before trying to edit it"
'       Exit Sub
'    End If
'
'    ingredientLocation = lstRecipieIngredients.listIndex + 1
'    'get list number. Remember that the Ingredient is going to be equal to the list Index +1
'    ingId = getIngredientID(ingredientLocation, selRecID)
'    otherValues = getOtherValue(ingId, selRecID)
'
'    If Len(Trim(otherValues)) > 0 Then
'       other = True
'    Else
'       other = False
'    End If
'    frmRecipieIngredientsEdit.Show
End Sub

Sub reloadForm()
   UserForm_Initialize
End Sub
Private Sub UserForm_Initialize()
'    sortRecipes
'    sortIngredients
'    sortMeassurements
   other = False

   selRecID = frmRecipieMain.recid
   'the first time the form is shown make sure that is readable but now editable
   'lockTextBoxes

   'get row number of selected recipie
   rowNumber = frmRecipieMain.rowNumber
   TRN = rowNumber
   'popullate text boxes
   popullateTextBoxes
   loadMeasurements
   loadIngredients
   lockInputBoxes

   'find all the ingredients related to the recipie and add them to the list
   findIngredients sRecipie.Cells(frmRecipieMain.rowNumber, 1).Value
End Sub

'Private Sub lockTextBoxes()
'
```

```
'    txtPreparationTime.Enabled = False
'    txtPreparationTime.BackColor = "&H80000004"
'
'    txtRecCountry.Enabled = False
'    txtRecCountry.BackColor = "&H80000004"
'
'    txtRecDescription.Enabled = False
'    txtRecDescription.BackColor = "&H80000004"
'
'    txtRecName.Enabled = False
'    txtRecName.BackColor = "&H80000004"
'
'    txtRecType.Enabled = False
'    txtRecType.BackColor = "&H80000004"
'
'    txtServingSize.Enabled = False
'    txtServingSize.BackColor = "&H80000004"
'
'    txtRecDirections.Enabled = False
'    txtRecDirections.BackColor = "&H80000004"
'
''    lstRecipieIngredients.Enabled = False
''    lstRecipieIngredients.BackColor = "&H80000004"
''
''    cmdAddIngredient.Enabled = False
''    cmdRemoveIngredient.Enabled = False
'End Sub
'
'Private Sub unlockTextBoxes()
'
'    txtPreparationTime.Enabled = True
'    txtPreparationTime.BackColor = "&H80000005"
'
'    txtRecCountry.Enabled = True
'    txtRecCountry.BackColor = "&H80000005"
'
'    txtRecDescription.Enabled = True
'    txtRecDescription.BackColor = "&H80000005"
'
'    txtRecName.Enabled = True
'    txtRecName.BackColor = "&H80000005"
'
'    txtRecType.Enabled = True
'    txtRecType.BackColor = "&H80000005"
'
'    txtServingSize.Enabled = True
'    txtServingSize.BackColor = "&H80000005"
'
'    txtRecDirections.Enabled = True
'    txtRecDirections.BackColor = "&H80000005"
'
''    lstRecipieIngredients.Enabled = True
''    lstRecipieIngredients.BackColor = "&H80000005"
''
''    cmdAddIngredient.Enabled = True
''    cmdRemoveIngredient.Enabled = True
'End Sub
```

```
Sub popullateTextBoxes()
   txtRecId = sRecipie.Cells(rowNumber, 1)
   txtRecName = sRecipie.Cells(rowNumber, 2)
   txtRecDescription = sRecipie.Cells(rowNumber, 3)
   txtRecDirections = sRecipie.Cells(rowNumber, 4)
   txtPreparationTime = sRecipie.Cells(rowNumber, 5)
   txtServingSize = sRecipie.Cells(rowNumber, 6)
   txtRecType = sRecipie.Cells(rowNumber, 7)
   txtRecCountry = sRecipie.Cells(rowNumber, 8)
End Sub

Sub findIngredients(recipiID As Long)
   Dim startingRow As Long
   Dim lstIndex As Long
   Dim condensedIngredientItem As String
   Dim ingredientQuantity As String
   Dim ingredientMeasurement As String
   Dim ingredienPreparation As String
   Dim ingredientName As String
   Dim ingredientOther As String

   startingRow = 2
   'get all the ingredients that match the recipi ID
   Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 2))) > 0
      If Trim(sRecipieIngredients.Cells(startingRow, 2)) = recipiID Then
         'get Ingredient Quantity
         ingredientQuantity = getIngredientQuantity(sRecipieIngredients.Cells(startingRow, 1), sRecipieIngredients.Cells(startingRow, 2))

         'get Measurement
         ingredientMeasurement = getMeasurementID(sRecipieIngredients.Cells(startingRow, 1), sRecipieIngredients.Cells(startingRow, 2))
         If LCase(Trim(ingredientMeasurement)) = "n/a" Then ingredientMeasurement = ""

         'get Ingrediet Preparation
         ingredienPreparation = getIngredientPreparation(sRecipieIngredients.Cells(startingRow, 1), sRecipieIngredients.Cells(startingRow, 2))

         'get Ingredient Name
         ingredientName = getIngredientName(sRecipieIngredients.Cells(startingRow, 1))
         If LCase(Trim(ingredientName)) = "n/a" Then ingredientName = ""

         If Len(Trim(sRecipieIngredients.Cells(startingRow, 6))) < 1 Then
            condensedIngredientItem = ingredientQuantity & " " & ingredientMeasurement & " " & ingredienPreparation & " " & ingredientName
         Else
            condensedIngredientItem = sRecipieIngredients.Cells(startingRow, 6)
         End If

         'add items to list
         lstRecipieIngredients.AddItem condensedIngredientItem
      End If
      startingRow = startingRow + 1
   Loop

End Sub

'get Ingredient Quantity
Function getIngredientQuantity(ingredientID As Long, recipieID As Long)
```

```
    Dim startingRow As Long
    startingRow = 2

    getIngredientQuantity = ""

    Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
        If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
            getIngredientQuantity = sRecipieIngredients.Cells(startingRow, 3)
            Exit Function
        End If
        startingRow = startingRow + 1
    Loop
End Function

'get Measurement
Function getMeasurementID(ingredientID As Long, recipieID As Long)
    Dim startingRow As Long
    Dim measurementId As Long

    startingRow = 2

    getMeasurementID = -1

    Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
        If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
            measurementId = sRecipieIngredients.Cells(startingRow, 5)
            getMeasurementID = measurementIDtoAbbreviation(measurementId)
            Exit Function
        End If
        startingRow = startingRow + 1
    Loop
End Function

Function measurementIDtoAbbreviation(measurementId As Long)
    Dim startingRow As Long
    startingRow = 2

    Do While Len(Trim(sMeassurement.Cells(startingRow, 1))) > 0
        If sMeassurement.Cells(startingRow, 1) = measurementId Then
            measurementIDtoAbbreviation = sMeassurement.Cells(startingRow, 3)
            Exit Function
        End If
        startingRow = startingRow + 1
    Loop
End Function

'get Ingrediet Preparation
Function getIngredientPreparation(ingredientID As Long, recipieID As Long)
    Dim startingRow As Long
    startingRow = 2

    getIngredientPreparation = ""

    Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
        If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
            getIngredientPreparation = sRecipieIngredients.Cells(startingRow, 4)
            Exit Function
```

```
      End If
      startingRow = startingRow + 1
   Loop
End Function


'get Ingredient Name
Function getIngredientName(ingredientID As Long)
   Dim startingRow As Long
   startingRow = 2
   getIngredientName = ""
   Do While Len(Trim(sIngredients.Cells(startingRow, 1))) > 0
      If sIngredients.Cells(startingRow, 1) = ingredientID Then
         getIngredientName = sIngredients.Cells(startingRow, 2)
         Exit Function
      End If
      startingRow = startingRow + 1
   Loop
End Function

Function getIngredientID(ingredientLocation As Long, recipieID As Long)
   Dim maxIterations As Long
   Dim startingRow As Long

   startingRow = 2

   Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
      If sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
         maxIterations = maxIterations + 1
      End If

      If maxIterations = ingredientLocation Then
         getIngredientID = sRecipieIngredients.Cells(startingRow, 1)
         Exit Do
      End If

      startingRow = startingRow + 1
   Loop

End Function

Function getOtherValue(ingredientID As Long, recipieID As Long)
   Dim maxIterations As Long
   Dim startingRow As Long

   startingRow = 2
   getOtherValue = ""

   Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
      If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
         getOtherValue = sRecipieIngredients.Cells(startingRow, 6)
         Exit Function
      End If
      startingRow = startingRow + 1
   Loop
End Function
```

```
Sub deleteIngredient(ingredientID As Long, recipieID As Long)
    Dim maxIterations As Long
    Dim startingRow As Long

    startingRow = 2

    Do While Len(Trim(sRecipieIngredients.Cells(startingRow, 1))) > 0
        If sRecipieIngredients.Cells(startingRow, 1) = ingredientID And sRecipieIngredients.Cells(startingRow, 2) = recipieID Then
            sRecipieIngredients.Cells(startingRow, 6).EntireRow.Delete
            Exit Sub
        End If
        startingRow = startingRow + 1
    Loop
End Sub

Function getRNwithRID(recid As Long)
    Dim rN As Long

    rN = 2
    Do While Len(Trim(sRecipie.Cells(rN, 1))) > 0
        If sRecipie.Cells(rN, 1) = recid Then
            getRNwithRID = rN
            Exit Do
        End If
        rN = rN + 1
    Loop
End Function

'Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
'    cmdClose_Click
'End Sub

Private Sub chkOther_Click()
    If chkOther.Value = True Then
        'lock the rest of the input boxes
        lockInputBoxes
    Else
        'unlock the rest of the input boxes
        unlockInputBoxes
    End If
End Sub

Sub lockInputBoxes()
    chkOther = True
    txtOther.Enabled = True
    txtOther.BackColor = "&H80000005"

    txtQty.Enabled = False
    txtQty.BackColor = "&H80000004"

    txtPreparation.Enabled = False
    txtPreparation.BackColor = "&H80000004"

    cmbMeasurements.Enabled = False
    cmbMeasurements.BackColor = "&H80000004"

    txtPreparation.BackColor = "&H80000004"
```

```
    cboIngredients.Enabled = False
    cboIngredients.BackColor = "&H80000004"

    txtPreparation.BackColor = "&H80000004"
End Sub

Sub unlockInputBoxes()
    chkOther = False
    txtOther.Enabled = False
    txtOther.BackColor = "&H80000004"

    txtQty.Enabled = True
    txtQty.BackColor = "&H80000005"

    txtPreparation.Enabled = True
    txtPreparation.BackColor = "&H80000005"

    cmbMeasurements.Enabled = True
    cmbMeasurements.BackColor = "&H80000005"
    txtPreparation.BackColor = "&H80000005"

    cboIngredients.Enabled = True
    cboIngredients.BackColor = "&H80000005"

    txtPreparation.BackColor = "&H80000005"
End Sub

'get Ingredient selected from combo box
Function getCmdIngredientID() As Long
    getCmdIngredientID = cboIngredients.listIndex
End Function

'get Measurement selected from combo box
Function getCmdMeassurementID() As Long
    getCmdMeassurementID = cmbMeasurements.listIndex
End Function

Sub loadMeasurements()
    Dim rowNum As Long
    Dim sellItem As Long
    rowNum = 2

    Do While Len(Trim(sMeassurement.Cells(rowNum, 1))) > 0
        cmbMeasurements.AddItem sMeassurement.Cells(rowNum, 2)
        rowNum = rowNum + 1
    Loop
End Sub

'load ingredients
Sub loadIngredients()
    Dim rowNum As Long
    Dim sellItem As Long
    rowNum = 2

    Do While Len(Trim(sIngredients.Cells(rowNum, 1))) > 0
        cboIngredients.AddItem sIngredients.Cells(rowNum, 2)
```

```
        rowNum = rowNum + 1
    Loop
End Sub

Sub clearTxtBoxes()
    txtOther = ""
    txtQty = ""
    txtPreparation.Enabled = True
    cmbMeasurements.listIndex = -1
    cboIngredients.listIndex = -1
End Sub
```

frmRecipieSearchI

```
Option Explicit

Dim resultsString As String
Dim resultsArrayRID() As String
Dim resultsArrayRN() As String
Dim resultsIndex As Long
Dim recipeWMNI() As String
Dim recipeIngredientsCMB() As String
Dim indexRecipeIngredientsCMB As Long
Dim recipeIngredientsTXT() As String
Dim IndexRecipeIngredientsTXT As Long
Dim recipeIngredientsTXT2() As String
Dim IndexRecipeIngredientsTXT2 As Long
Dim recipeIngredientsTXT3() As String
Dim IndexRecipeIngredientsTXT3 As Long


Private Sub cmbIngredient1_Change()
    If cmbIngredient1.listIndex >= 0 Then
        cmbIngredient2.Enabled = True
        cmbIngredient2.BackColor = &H80000005
    End If
End Sub

Private Sub cmbIngredient2_Change()
    If cmbIngredient2.listIndex = cmbIngredient1.listIndex Then
        MsgBox "You have to select different ingredients"
        cmbIngredient2.listIndex = -1
        Exit Sub
    Else
        If cmbIngredient1.listIndex >= 0 And cmbIngredient2.listIndex >= 0 Then
            cmbIngredient3.Enabled = True
            cmbIngredient3.BackColor = &H80000005
        End If
    End If
End Sub

Private Sub cmbIngredient3_Change()

    If cmbIngredient3.listIndex = cmbIngredient2.listIndex Or cmbIngredient3.listIndex = cmbIngredient1.listIndex Then
        MsgBox "You have to select different ingredients"
        cmbIngredient3.listIndex = -1
        Exit Sub
    Else
```

```
      If cmbIngredient2.listIndex < 0 And cmbIngredient1.listIndex < 0 Then
         cmbIngredient3.listIndex = -1
         cmbIngredient3.Enabled = False
         cmbIngredient3.BackColor = &H80000004
      End If
   End If

End Sub


Private Sub cmdClearSearch_Click()
   cmbIngredient1.listIndex = -1

   cmbIngredient2.listIndex = -1
   cmbIngredient2.Enabled = False
   cmbIngredient2.BackColor = &H80000004

   cmbIngredient3.listIndex = -1
   cmbIngredient3.Enabled = False
   cmbIngredient3.BackColor = &H80000004

   txtIngredient4 = ""
   txtIngredient5 = ""
   txtIngredient6 = ""

   Do While lstResults.ListCount > 0
      lstResults.RemoveItem 0
   Loop

End Sub

'Private Sub cmdClose_Click()
'    Unload Me
'End Sub



Private Sub cmdSearch_Click()
   Dim ingredient1 As Long
   Dim ingredient2 As Long
   Dim ingredient3 As Long
   Dim ingredient4 As String
   Dim ingredient5 As String
   Dim ingredient6 As String
   Dim rowNumber As Long
   Dim numberOfIngredients As Long
   Dim numberOfIngredientsFromCombo As Long
   Dim numberOfIngredientsFromTxt As Long
   Dim tempRID As Long

   rowNumber = 2

   'get text from your text boxes
   ingredient1 = CLng(cmbIngredient1.listIndex)
   ingredient2 = CLng(cmbIngredient2.listIndex)
   ingredient3 = CLng(cmbIngredient3.listIndex)
   ingredient4 = Trim(txtIngredient4)
```

```
    ingredient5 = Trim(txtIngredient5)
    ingredient6 = Trim(txtIngredient6)

    If ingredient1 < 0 And ingredient2 < 0 And ingredient3 < 0 And Len(ingredient4) < 1 And Len(ingredient5) < 1 And Len(ingredient6) < 1 Then
        MsgBox "Specify at least one search criteria", vbCritical
        Exit Sub
    End If

    Do While lstResults.ListCount > 0
        lstResults.RemoveItem 0
    Loop

    'determine how many ingredients need to be found
    If ingredient1 >= 0 Then
        numberOfIngredients = numberOfIngredients + 1
        numberOfIngredientsFromCombo = numberOfIngredientsFromCombo + 1
        ingredient1 = sIngredients.Cells(ingredient1 + 2, 1)

        ReDim Preserve recipeIngredientsCMB(indexRecipeIngredientsCMB)
        recipeIngredientsCMB(indexRecipeIngredientsCMB) = ingredient1
        recipeIngredientsCMB(indexRecipeIngredientsCMB) = ingredient1

        indexRecipeIngredientsCMB = indexRecipeIngredientsCMB + 1
    End If

    If ingredient2 >= 0 Then
        numberOfIngredients = numberOfIngredients + 1
        numberOfIngredientsFromCombo = numberOfIngredientsFromCombo + 1

        ingredient2 = sIngredients.Cells(ingredient2 + 2, 1)

        ReDim Preserve recipeIngredientsCMB(indexRecipeIngredientsCMB)
        recipeIngredientsCMB(indexRecipeIngredientsCMB) = ingredient2
        indexRecipeIngredientsCMB = indexRecipeIngredientsCMB + 1
    End If

    If ingredient3 >= 0 Then
        numberOfIngredientsFromCombo = numberOfIngredientsFromCombo + 1
        numberOfIngredients = numberOfIngredients + 1
        ingredient3 = sIngredients.Cells(ingredient3 + 2, 1)

        ReDim Preserve recipeIngredientsCMB(indexRecipeIngredientsCMB)
        recipeIngredientsCMB(indexRecipeIngredientsCMB) = ingredient3
        indexRecipeIngredientsCMB = indexRecipeIngredientsCMB + 1
    End If

'Dim counter As Long
'For counter = 0 To UBound(recipeIngredientsCMB)
'    Debug.Print "from combo " & counter & ": " & recipeIngredientsCMB(counter)
'Next


    'Find recipes with Minimun ingredient required
    Do While Len(Trim(sRecipie.Cells(rowNumber, 1))) > 0
        Dim wasFound As Boolean

        tempRID = sRecipie.Cells(rowNumber, 1)
```

```
    wasFound = findRID(tempRID)

    If wasFound = True Then
        sRecipieIngredients.Select
        Columns("B:B").Select
            Selection.find(What:=tempRID, After:=ActiveCell, LookIn:=xlFormulas, _
            LookAt:=xlWhole, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
            MatchCase:=False, SearchFormat:=False).Activate

            Dim numberOI As Long
            Dim indexRecipeWMNI As Long

            Do While (tempRID = Trim(ActiveCell))
                numberOI = numberOI + 1
                ActiveCell.Offset(1).Activate

                If numberOI >= numberOfIngredients Then
                    'add the recipeID selected as a slected candidate for the next search
                    ReDim Preserve recipeWMNI(indexRecipeWMNI)
                    recipeWMNI(indexRecipeWMNI) = tempRID
                    indexRecipeWMNI = indexRecipeWMNI + 1
                    Exit Do
                End If
            Loop
    End If
    rowNumber = rowNumber + 1
Loop


Dim maxValues As Long
Dim iterator As Long
Dim isThere As Boolean
Dim minIngredientsMet As Boolean

If Trim(sRecipieIngredients.Cells(2, 1)) = "" Or Trim(sRecipie.Cells(2, 1)) = "" Then
    MsgBox "There are no recipes with the specified criteria"
    Exit Sub
Else
    maxValues = sRecipieIngredients.Cells((sRecipieIngredients.Cells(1, 1).End(xlDown).row + 1), 1).row
End If

If maxValues >= 2 Then
    If Len(ingredient4) > 0 Then
        numberOfIngredientsFromTxt = numberOfIngredientsFromTxt + 1
        If iterator = 0 Then iterator = 2
        'search Other and find the recipe ID
        Do While iterator < maxValues
            If Len(Trim(sRecipieIngredients.Cells(iterator, 6))) > 0 Then
                If InStr(1, LCase(Trim(sRecipieIngredients.Cells(iterator, 6))), LCase(ingredient4)) > 0 Then
                    minIngredientsMet = recipeHasMinimumIngredients(sRecipieIngredients.Cells(iterator, 2))
                    If minIngredientsMet = True Then
                        isThere = seelfIngredient4Exists(sRecipieIngredients.Cells(iterator, 1))
                        If isThere = False Then
                            ReDim Preserve recipeIngredientsTXT(IndexRecipeIngredientsTXT)
                            recipeIngredientsTXT(IndexRecipeIngredientsTXT) = sRecipieIngredients.Cells(iterator, 1)
                            IndexRecipeIngredientsTXT = IndexRecipeIngredientsTXT + 1
```

```
                numberOfIngredients = numberOfIngredients + 1
              End If
            End If

          End If
        End If
        iterator = iterator + 1
    Loop
    iterator = 0
    isThere = False
    minIngredientsMet = False
End If

If Len(ingredient5) > 0 Then
    numberOfIngredientsFromTxt = numberOfIngredientsFromTxt + 1
    If iterator = 0 Then iterator = 2
    'search Other and find the recipe ID
    Do While iterator < maxValues
        If Len(Trim(sRecipieIngredients.Cells(iterator, 6))) > 0 Then
            If InStr(1, LCase(Trim(sRecipieIngredients.Cells(iterator, 6))), LCase(ingredient5)) > 0 Then
                minIngredientsMet = recipeHasMinimumIngredients(sRecipieIngredients.Cells(iterator, 2))
                If minIngredientsMet = True Then
                    isThere = seelfIngredient5Exists(sRecipieIngredients.Cells(iterator, 1))
                    If isThere = False Then
                        ReDim Preserve recipeIngredientsTXT2(IndexRecipeIngredientsTXT2)
                        recipeIngredientsTXT2(IndexRecipeIngredientsTXT2) = sRecipieIngredients.Cells(iterator, 1)
                        IndexRecipeIngredientsTXT2 = IndexRecipeIngredientsTXT2 + 1
                        numberOfIngredients = numberOfIngredients + 1
                    End If
                End If
            End If
        End If
        iterator = iterator + 1
    Loop
    iterator = 0
    isThere = False
    minIngredientsMet = False
End If

If Len(ingredient6) > 0 Then
    numberOfIngredientsFromTxt = numberOfIngredientsFromTxt + 1

    If iterator = 0 Then iterator = 2
    'search Other and find the recipe ID
    Do While iterator < maxValues
        If Len(Trim(sRecipieIngredients.Cells(iterator, 6))) > 0 Then
            If InStr(1, LCase(Trim(sRecipieIngredients.Cells(iterator, 6))), LCase(ingredient6)) > 0 Then
                minIngredientsMet = recipeHasMinimumIngredients(sRecipieIngredients.Cells(iterator, 2))
                If minIngredientsMet = True Then
                    isThere = seelfIngredient6Exists(sRecipieIngredients.Cells(iterator, 1))
                    If isThere = False Then
                        ReDim Preserve recipeIngredientsTXT3(IndexRecipeIngredientsTXT3)
                        recipeIngredientsTXT3(IndexRecipeIngredientsTXT3) = sRecipieIngredients.Cells(iterator, 1)
                        IndexRecipeIngredientsTXT3 = IndexRecipeIngredientsTXT3 + 1
                        numberOfIngredients = numberOfIngredients + 1
                    End If
                End If
```

```
            End If
          End If
          iterator = iterator + 1
        Loop
        iterator = 0
        isThere = False
        minIngredientsMet = False
      End If
    End If

'Dim counter2 As Long
'For counter2 = 0 To UBound(recipeIngredientsTXT)
'    Debug.Print "from txt " & counter2 & ": " & recipeIngredientsTXT(counter2)
'Next

    Dim numberFoundCombo As Long
    Dim numberFoundText As Long
    Dim numberFoundComboText As Long
    Dim found4 As Boolean
    Dim found5 As Boolean
    Dim found6 As Boolean
    Dim rN As Long

    If numberOfIngredientsFromCombo > 0 And numberOfIngredientsFromTxt = 0 Then
      rowNumber = 2
      rN = 2
      Do While Len(sRecipie.Cells(rowNumber, 1)) > 0
        If recipeHasMinimumIngredients(sRecipie.Cells(rowNumber, 1)) = True Then
          Do While sRecipieIngredients.Cells(rN, 1) > 0
            If Trim(sRecipieIngredients.Cells(rN, 2)) = Trim(sRecipie.Cells(rowNumber, 1)) Then
              If ingredient1 > 0 Then
                If ingredient1 = sRecipieIngredients.Cells(rN, 1) Then
                  numberFoundCombo = numberFoundCombo + 1
                  If numberFoundCombo >= numberOfIngredientsFromCombo Then
                    ReDim Preserve resultsArrayRID(resultsIndex)
                    ReDim Preserve resultsArrayRN(resultsIndex)

                    resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
                    resultsArrayRN(resultsIndex) = rowNumber
                    lstResults.AddItem sRecipie.Cells(rowNumber, 2)
                    resultsIndex = resultsIndex + 1
                    Exit Do
                  End If
                End If
              End If

              If ingredient2 > 0 Then
                If ingredient2 = sRecipieIngredients.Cells(rN, 1) Then
                  numberFoundCombo = numberFoundCombo + 1
                  If numberFoundCombo >= numberOfIngredientsFromCombo Then
'                     ReDim Preserve resultsArrayRID(resultsIndex)
'                     resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
'                     resultsIndex = resultsIndex + 1
                    ReDim Preserve resultsArrayRID(resultsIndex)
                    ReDim Preserve resultsArrayRN(resultsIndex)

                    resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
```

```
                    resultsArrayRN(resultsIndex) = rowNumber

                    lstResults.AddItem sRecipie.Cells(rowNumber, 2)
                    resultsIndex = resultsIndex + 1

                    resultsIndex = resultsIndex + 1
                    Exit Do
                  End If
                End If
              End If

              If ingredient3 > 0 Then
                If ingredient3 = sRecipieIngredients.Cells(rN, 1) Then
                  numberFoundCombo = numberFoundCombo + 1
                  If numberFoundCombo >= numberOfIngredientsFromCombo Then
'                   ReDim Preserve resultsArrayRID(resultsIndex)
'                   resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
'                   resultsIndex = resultsIndex + 1
                    ReDim Preserve resultsArrayRID(resultsIndex)
                    ReDim Preserve resultsArrayRN(resultsIndex)

                    resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
                    resultsArrayRN(resultsIndex) = rowNumber
                    lstResults.AddItem sRecipie.Cells(rowNumber, 2)

                    resultsIndex = resultsIndex + 1
                    Exit Do
                  End If
                End If
              End If
            End If
            rN = rN + 1
          Loop
        End If
        rN = 2
        rowNumber = rowNumber + 1
      Loop
      'Up to here is working fine. At least for the combo box part
      '-------------------------------------------------------------------------------------------------------------------
    ElseIf numberOfIngredientsFromCombo = 0 And numberOfIngredientsFromTxt > 0 Then
      rowNumber = 2
      rN = 2
      iterator = 0
      Do While Len(sRecipie.Cells(rowNumber, 1)) > 0
        If recipeHasMinimumIngredients(sRecipie.Cells(rowNumber, 1)) = True Then


'this
'is
'where
'you
'stoped
```

```
        Do While sRecipieIngredients.Cells(rN, 1) > 0
          If LCase(sRecipieIngredients.Cells(rN, 2)) = LCase(sRecipie.Cells(rowNumber, 1)) Then
            If Len(ingredient4) > 0 Then
              On Error GoTo 10
              For iterator = 0 To UBound(recipeIngredientsTXT)
                If sRecipieIngredients.Cells(rN, 1) = recipeIngredientsTXT(iterator) Then
                  numberFoundText = numberFoundText + 1
                  'This can give false information.I need another variable
                  If numberFoundText >= numberOfIngredientsFromTxt Then
                    If numberFoundText <= numberOfIngredients Then




                      ReDim Preserve resultsArrayRID(resultsIndex)
                      ReDim Preserve resultsArrayRN(resultsIndex)

                      resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
                      resultsArrayRN(resultsIndex) = rowNumber
                      lstResults.AddItem sRecipie.Cells(rowNumber, 2)

                      resultsIndex = resultsIndex + 1
                      Exit Do
                    End If
                  End If
                End If
              Next
            End If

10:

            If Len(ingredient5) > 0 Then
              On Error GoTo 11
              For iterator = 0 To UBound(recipeIngredientsTXT2)
                If LCase(sRecipieIngredients.Cells(rN, 1)) = LCase(recipeIngredientsTXT2(iterator)) Then
                  numberFoundText = numberFoundText + 1
                  'This can give false information.I need another variable
                  If numberFoundText >= numberOfIngredientsFromTxt Then
                    If numberFoundText >= numberOfIngredients Then
'                     ReDim Preserve resultsArrayRID(resultsIndex)
'                     resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
'                     resultsIndex = resultsIndex + 1
                      ReDim Preserve resultsArrayRID(resultsIndex)
                      ReDim Preserve resultsArrayRN(resultsIndex)

                      resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
                      resultsArrayRN(resultsIndex) = rowNumber
                      lstResults.AddItem sRecipie.Cells(rowNumber, 2)

                      resultsIndex = resultsIndex + 1
                      Exit Do
                    End If
                  End If
                End If
              Next
```

```
            End If
11:
            On Error GoTo 12
            If Len(ingredient6) > 0 Then

               For iterator = 0 To UBound(recipeIngredientsTXT3)
                  If LCase(sRecipieIngredients.Cells(rN, 1)) = LCase(recipeIngredientsTXT3(iterator)) Then
                     numberFoundText = numberFoundText + 1
                     'This can give false information.I need another variable
                     If numberFoundText >= numberOfIngredientsFromTxt Then
                        If numberFoundText >= numberOfIngredients Then
'                          ReDim Preserve resultsArrayRID(resultsIndex)
'                          resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
'                          resultsIndex = resultsIndex + 1
                           ReDim Preserve resultsArrayRID(resultsIndex)
                           ReDim Preserve resultsArrayRN(resultsIndex)

                           resultsArrayRID(resultsIndex) = sRecipie.Cells(rowNumber, 1)
                           resultsArrayRN(resultsIndex) = rowNumber
                           lstResults.AddItem sRecipie.Cells(rowNumber, 2)

                           resultsIndex = resultsIndex + 1
                           Exit Do
                        End If
                     End If
                  End If
               Next
            End If
12:
         End If

         rN = rN + 1
      Loop
   End If
   rN = 2
   rowNumber = rowNumber + 1
 Loop
ElseIf numberOfIngredientsFromCombo > 0 And numberOfIngredientsFromTxt > 0 Then
'good luck

End If

End Sub


Private Sub imgExit_Click()
   Unload Me
End Sub

Private Sub imgSearch_Click()
   cmdSearch_Click
End Sub

Private Sub lstResults_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
   Dim foundRecipeID As String
   Dim foundRowNumber As String
```

```
    If lstResults.ListCount = 0 Or lstResults.listIndex < 0 Then
      MsgBox "An Recipe must be selected before trying to view it"
      Exit Sub
    End If

    foundRecipeID = resultsArrayRID(lstResults.listIndex)
    foundRowNumber = resultsArrayRN(lstResults.listIndex)

    frmRecipieMain.recid = foundRecipeID
    frmRecipieMain.rowNumber = foundRowNumber

    frmRecipieNTFound.Show

End Sub

Private Function findRID(RID As Long) As Boolean
    findRID = False
    On Error GoTo 1
    sRecipieIngredients.Select
    Columns("B:B").Select
    Selection.find(What:=RID, After:=ActiveCell, LookIn:=xlFormulas, _
        LookAt:=xlWhole, SearchOrder:=xlByRows, SearchDirection:=xlNext, _
        MatchCase:=False, SearchFormat:=False).Activate
        findRID = True
        GoTo 2
1:
    findRID = False
    Exit Function
2:

End Function



Private Sub UserForm_Initialize()
    Dim ingRN As Long

    'Sort Recipe/Ingredients
    sortRecipeIngrediets

    ingRN = 2
    'load the 5 combo boxes with the ingredients information
    sIngredients.Select
    Do While (Len(Trim(sIngredients.Cells(ingRN, 2))) > 0)
        cmbIngredient1.AddItem (Trim(sIngredients.Cells(ingRN, 2)))
        cmbIngredient2.AddItem (Trim(sIngredients.Cells(ingRN, 2)))
        cmbIngredient3.AddItem (Trim(sIngredients.Cells(ingRN, 2)))
        ingRN = ingRN + 1
    Loop

End Sub

Function getArraySize(a As Variant)
    Dim aSize As Long

    On Error GoTo 1
        getArraySize = UBound(a)
```

```
    Exit Function

1:
    getArraySize = -1
End Function


Function selRecipeExists(ByVal recipeID As Long) As Boolean
    Dim i As Long
    On Error GoTo 1:

    For i = 0 To UBound(resultsArrayRID)
        If recipeID = resultsArrayRID(i) Then selRecipeExists = True
        If selRecipeExists = True Then Exit Function
    Next
    Exit Function
1:
    selRecipeExists = False
End Function


Function seelfIngredient4Exists(ingredietID As Long) As Boolean
    Dim i As Long
    On Error GoTo 1:

    For i = 0 To UBound(recipeIngredientsTXT)
        If ingredietID = recipeIngredientsTXT(i) Then seelfIngredient4Exists = True
        If seelfIngredient4Exists = True Then Exit Function
    Next
    Exit Function
1:
    seelfIngredient4Exists = False
End Function

Function seelfIngredient5Exists(ingredietID As Long) As Boolean
    Dim i As Long
    On Error GoTo 1:

    For i = 0 To UBound(recipeIngredientsTXT2)
        If ingredietID = recipeIngredientsTXT2(i) Then seelfIngredient5Exists = True
        If seelfIngredient5Exists = True Then Exit Function
    Next
    Exit Function
1:
    seelfIngredient5Exists = False
End Function

Function seelfIngredient6Exists(ingredietID As Long) As Boolean
    Dim i As Long
    On Error GoTo 1:

    For i = 0 To UBound(recipeIngredientsTXT3)
        If ingredietID = recipeIngredientsTXT3(i) Then seelfIngredient6Exists = True
        If seelfIngredient6Exists = True Then Exit Function
    Next
    Exit Function
1:
```

```
    seelfIngredient6Exists = False
End Function

Function recipeHasMinimumIngredients(recipeID As Long) As Boolean
    Dim i As Long
    On Error GoTo 1:

    For i = 0 To UBound(recipeWMNI)
        If recipeID = recipeWMNI(i) Then recipeHasMinimumIngredients = True
        If recipeHasMinimumIngredients = True Then Exit Function
    Next
    Exit Function
1:
    recipeHasMinimumIngredients = False
End Function

Function isInRecipieIngredientsCMB(ingid As Long) As Boolean
    Dim i As Long
    On Error GoTo 1:

    For i = 0 To UBound(recipeIngredientsCMB)
        If ingid = recipeIngredientsCMB(i) Then isInRecipieIngredientsCMB = True
        If isInRecipieIngredientsCMB = True Then Exit Function
    Next
    Exit Function
1:
    isInRecipieIngredientsCMB = False
End Function
```

frmRecipieSearchNT

```
Option Explicit

Dim resultsString As String
Dim resultsArrayRID() As String
Dim resultsArrayRN() As String
Dim resultsIndex As Long


Private Sub cmdClearSearch_Click()
    txtRecName = ""
    txtRecType = ""
    txtRecCountry = ""
    txtPreparationTime = ""
    txtServingSize = ""

    Do While lstResults.ListCount > 0
        lstResults.RemoveItem 0
    Loop

End Sub

'Private Sub cmdClose_Click()
'    Unload Me
'End Sub

Private Sub cmdSearch_Click()
    Dim recName As String
```

```
    Dim recType As String
    Dim recCountry As String
    Dim recPreparationTime As String
    Dim recServingSize As String
    Dim rowNumber As Long


    recName = Trim(txtRecName)
    recType = Trim(txtRecType)
    recCountry = Trim(txtRecCountry)
    recPreparationTime = Trim(txtPreparationTime)
    recServingSize = Trim(txtServingSize)

    Do While lstResults.ListCount > 0
        lstResults.RemoveItem 0
    Loop

    If Trim(recName & recType & recCountry & recPreparationTime & recServingSize) = "" Then
        MsgBox "Specify at least one search criteria", vbCritical
        Exit Sub
    End If

    rowNumber = 2

    Do Until Trim(sRecipie.Cells(rowNumber, 1)) = ""

        If Trim(sRecipie.Cells(rowNumber, 2)) = "" Then sRecipie.Cells(rowNumber, 2) = " "
        If Trim(sRecipie.Cells(rowNumber, 7)) = "" Then sRecipie.Cells(rowNumber, 7) = " "
        If Trim(sRecipie.Cells(rowNumber, 8)) = "" Then sRecipie.Cells(rowNumber, 8) = " "
        If Trim(sRecipie.Cells(rowNumber, 5)) = "" Then sRecipie.Cells(rowNumber, 5) = " "
        If Trim(sRecipie.Cells(rowNumber, 6)) = "" Then sRecipie.Cells(rowNumber, 6) = " "


        If InStr(1, LCase(sRecipie.Cells(rowNumber, 2)), LCase(recName)) > 0 And _
            InStr(1, LCase(sRecipie.Cells(rowNumber, 7)), LCase(recType)) > 0 And _
            InStr(1, LCase(sRecipie.Cells(rowNumber, 8)), LCase(recCountry)) > 0 And _
            InStr(1, LCase(sRecipie.Cells(rowNumber, 5)), LCase(recPreparationTime)) > 0 And _
            InStr(1, LCase(sRecipie.Cells(rowNumber, 6)), LCase(recServingSize)) > 0 Then

                ReDim Preserve resultsArrayRID(resultsIndex)
                ReDim Preserve resultsArrayRN(resultsIndex)

                resultsArrayRID(resultsIndex) = Trim(sRecipie.Cells(rowNumber, 1))
                resultsArrayRN(resultsIndex) = rowNumber

                lstResults.AddItem sRecipie.Cells(rowNumber, 2)
                resultsIndex = resultsIndex + 1

        End If

        rowNumber = rowNumber + 1

    Loop

End Sub
```

```
Private Sub imgExit_Click()
   Unload Me
End Sub

Private Sub imgSearch_Click()
   cmdSearch_Click
End Sub

Private Sub lstResults_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
   Dim foundRecipeID As String
   Dim foundRowNumber As String

   If lstResults.ListCount = 0 Or lstResults.listIndex < 0 Then
      MsgBox "An Recipe must be selected before trying to view it"
      Exit Sub
   End If

   foundRecipeID = resultsArrayRID(lstResults.listIndex)
   foundRowNumber = resultsArrayRN(lstResults.listIndex)

   frmRecipieMain.recid = foundRecipeID
   frmRecipieMain.rowNumber = foundRowNumber

   frmRecipieNTFound.Show

End Sub

Private Sub UserForm_Click()

End Sub
```

Module1

```
Option Explicit


Public Function URLEncode(StringToEncode As String, Optional _
  UsePlusRatherThanHexForSpace As Boolean = False) As String

Dim TempAns As String
Dim CurChr As Integer
CurChr = 1
Do Until CurChr - 1 = Len(StringToEncode)
 Select Case Asc(mId(StringToEncode, CurChr, 1))
   Case 48 To 57, 65 To 90, 97 To 122
     TempAns = TempAns & mId(StringToEncode, CurChr, 1)
   Case 32
     If UsePlusRatherThanHexForSpace = True Then
       TempAns = TempAns & "+"
     Else
       TempAns = TempAns & "%" & Hex(32)
     End If
  Case Else
       TempAns = TempAns & "%" & _
           Format(Hex(Asc(mId(StringToEncode, _
           CurChr, 1))), "00")
End Select
```

```
  CurChr = CurChr + 1
Loop

URLEncode = TempAns
End Function


Public Function URLDecode(StringToDecode As String) As String

Dim TempAns As String
Dim CurChr As Integer

CurChr = 1

Do Until CurChr - 1 = Len(StringToDecode)
  Select Case mId(StringToDecode, CurChr, 1)
    Case "+"
      TempAns = TempAns & " "
    Case "%"
      TempAns = TempAns & Chr(Val("&h" & _
        mId(StringToDecode, CurChr + 1, 2)))
      CurChr = CurChr + 2
    Case Else
      TempAns = TempAns & mId(StringToDecode, CurChr, 1)
  End Select

CurChr = CurChr + 1
Loop

URLDecode = TempAns
End Function

Public Sub sortRecipes()
    sRecipie.Select
    sRecipie.Range("A:H").Select
    ActiveWorkbook.Worksheets("Recipe").Sort.SortFields.Clear
    ActiveWorkbook.Worksheets("Recipe").Sort.SortFields.add Key:=Range("B2:" & sRecipie.Range("B2").End(xlDown).address) _
        , SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    With ActiveWorkbook.Worksheets("Recipe").Sort
        .SetRange Range("A:H")
        .Header = xlYes
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With

End Sub

Public Sub sortIngredients()
    sIngredients.Select
    sIngredients.Range("A:B").Select
    ActiveWorkbook.Worksheets("Ingredients").Sort.SortFields.Clear
    ActiveWorkbook.Worksheets("Ingredients").Sort.SortFields.add Key:=Range("B2:" & sIngredients.Range("B2").End(xlDown).address) _
        , SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    With ActiveWorkbook.Worksheets("Ingredients").Sort
        .SetRange Range("A:B")
```

```
      .Header = xlYes
      .MatchCase = False
      .Orientation = xlTopToBottom
      .SortMethod = xlPinYin
      .Apply
   End With
End Sub


Public Sub sortRecipeIngrediets()
   sRecipieIngredients.Select
   Columns("A:F").Select
   ActiveWorkbook.Worksheets("RecipeIngredients").Sort.SortFields.Clear
   ActiveWorkbook.Worksheets("RecipeIngredients").Sort.SortFields.add Key:=Range("B2:" & sRecipie.Range("B2").End(xlDown).address) _
      , SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
   With ActiveWorkbook.Worksheets("RecipeIngredients").Sort
      .SetRange Range("A:F")
      .Header = xlYes
      .MatchCase = False
      .Orientation = xlTopToBottom
      .SortMethod = xlPinYin
      .Apply
   End With
End Sub

Public Sub sortMeassurements()
   sMeassurement.Select
   sMeassurement.Range("A:B").Select
   ActiveWorkbook.Worksheets("Recipe").Sort.SortFields.Clear
   ActiveWorkbook.Worksheets("Recipe").Sort.SortFields.add Key:=Range("B2:" & sMeassurement.Range("B2").End(xlDown).address) _
      , SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
   With ActiveWorkbook.Worksheets("Recipe").Sort
      .SetRange Range("A:B")
      .Header = xlYes
      .MatchCase = False
      .Orientation = xlTopToBottom
      .SortMethod = xlPinYin
      .Apply
   End With
End Sub


Public Function get3RandomNumbers()
   Dim first As String
   Dim second As String
   Dim third As String

   first = WorksheetFunction.RandBetween(0, 9)
   second = WorksheetFunction.RandBetween(0, 9)
   third = WorksheetFunction.RandBetween(0, 9)
   get3RandomNumbers = first & second & third

End Function

Public Function get6RandomNumbers()
   Dim first As String
   Dim second As String
```

```
    Dim third As String
    Dim fourth As String
    Dim fifth As String
    Dim sixth As String
'   Dim seventh As String
'   Dim eigth As String

    first = WorksheetFunction.RandBetween(1, 9)
    second = WorksheetFunction.RandBetween(0, 9)
    third = WorksheetFunction.RandBetween(0, 9)
    fourth = WorksheetFunction.RandBetween(0, 9)
    fifth = WorksheetFunction.RandBetween(0, 9)
    sixth = WorksheetFunction.RandBetween(0, 9)
'   seveth = WorksheetFunction.RandBetween(0, 9)
'   eigth = WorksheetFunction.RandBetween(0, 9)

    get6RandomNumbers = first & second & third & fourth & fifth & sixth '& seventh & eigth

End Function

Public Sub deleteRecipie(RID As Long, rtD As Long)
    Dim bC As Long

    bC = 2
    'delete all ingredients
    Do While Len(Trim(sRecipieIngredients.Cells(bC, 2))) > 0

        'Debug.Print "the value of rec id is: " & sRecipieIngredients.Cells(bC, 2) & " and the value of the passed IRC is " & rtD
        If LCase(Trim(sRecipieIngredients.Cells(bC, 2))) = LCase(RID) Then
            sRecipieIngredients.Cells(bC, 2).EntireRow.Delete
            bC = bC - 1
        End If
        bC = bC + 1
    Loop

    'delte the recipie
    sRecipie.Cells(rtD, 1).EntireRow.Delete
End Sub


Function readFile(path As String) As String
  ' returns the contents of a text file
  Open path For Input As #1
    readFile = Input(LOF(1), #1)
  Close #1
End Function




Function sendGMail(sendTo As String, from As String, pw As String, subject As String, body As String, Optional attachment As String) As Boolean

    Dim iMsg As Object
    Dim iConf As Object
    Dim schema As String

    Set iMsg = CreateObject("CDO.Message")
```

```
    Set iConf = CreateObject("CDO.Configuration")


    ' send one copy with Google SMTP server (with autentication)
    schema = "http://schemas.microsoft.com/cdo/configuration/"
    iConf.Fields.Item(schema & "sendusing") = 2
    iConf.Fields.Item(schema & "smtpserver") = "smtp.gmail.com"
    iConf.Fields.Item(schema & "smtpserverport") = 465
    iConf.Fields.Item(schema & "smtpauthenticate") = 1
    iConf.Fields.Item(schema & "sendusername") = from
    iConf.Fields.Item(schema & "sendpassword") = pw
    iConf.Fields.Item(schema & "smtpusessl") = 1
    iConf.Fields.Update


    iMsg.To = sendTo
    iMsg.from = from
    iMsg.subject = subject
    '.HTMLBody = body
    iMsg.TextBody = body
    iMsg.Sender = "Myname"
    iMsg.Organization = "Myname"
    iMsg.ReplyTo = from

    If attachment > "" Then
      iMsg.AddAttachment attachment
    End If

    Set iMsg.Configuration = iConf
    On Error Resume Next
     iMsg.Send
     If Err.Number = 0 Then
       sendGMail = True
     Else
       sendGMail = False
     End If
    On Error GoTo 0

    Set iMsg = Nothing
    Set iConf = Nothing

End Function



Function ingredientInRecipe(ingid As Long, recid As Long) As Boolean
    Dim bR As Long
    bR = 2
    ingredientInRecipe = False
    Do While Len(Trim(sRecipieIngredients.Cells(bR, 1))) > 0
        If Trim(sRecipieIngredients.Cells(bR, 1)) = ingid And Trim(sRecipieIngredients.Cells(bR, 2)) = recid Then
            ingredientInRecipe = True
            Exit Do
        End If
        bR = bR + 1
    Loop
End Function
```

ribbonx

```
'Callback for customButton1 onAction
Sub startRecipeOrganizer(control As IRibbonControl)
  frmRecipieMain.Show
End Sub
```

agent

```
  Dim ie As Object
  Dim theHTML As String
  Dim pos As Long
  Dim region As String
' These three documents are the ones I use to understand the IE object
' InternetExplorere object reference: http://msdn2.microsoft.com/en-us/library/Aa752084.aspx#
' Document object recerence:         http://msdn2.microsoft.com/en-us/library/ms531073.aspx
' body object reference:             http://msdn2.microsoft.com/en-us/library/ms535205.aspx#

Public Property Get visible() As Boolean
  visible = ie.visible
End Property

Public Property Let visible(theValue As Boolean)
  ie.visible = theValue
End Property


Public Sub updateHTML()
  theHTML = documentSource()
End Sub

Public Function documentSource()


  Dim x As Long

'  For x = 0 To ie.document.all.Length
'    If ie.document.all(x).tagname = "HTML" Then
'      documentSource = ie.document.all(x).outerHTML
'      Exit Function
'    End If
'  Next

documentSource = ie.document.body.outerHTML


End Function


Public Property Get text() As String
  text = theHTML
End Property

Public Property Let text(theValue As String)
  theHTML = theValue
End Property

Public Property Get position() As Long
  position = pos
```

```
End Property

Public Property Let position(theValue As Long)
  pos = theValue
  If pos < 1 Then pos = 1
End Property

Public Property Get explorer() As Object
  Set explorer = ie
End Property

Sub initializeIE()
    'call this subprocedure to start internet explorer up
    Set ie = CreateObject("internetexplorer.application")
    pos = 1
End Sub

Sub terminateIE()
  ' call this subprocedure when you are finished with IE to close it down
  ie.Quit
  Set ie = Nothing
End Sub

Sub saveImage(url As String, path As String, Optional cookie As String)
' path is full path including filename
    strFileURL = url
    strHDLocation = path

' Fetch the file
    Set objXMLHTTP = CreateObject("MSXML2.XMLHTTP")

    objXMLHTTP.Open "GET", strFileURL, False
    If cookie > "" Then
      objXMLHTTP.setRequestHeader "Cookie", cookie
      objXMLHTTP.setRequestHeader "Cookie", cookie
    End If
    objXMLHTTP.Send

    If objXMLHTTP.Status = 200 Then
      Set objADOStream = CreateObject("ADODB.Stream")
      objADOStream.Open
      objADOStream.Type = 1 'adTypeBinary

      objADOStream.Write objXMLHTTP.responsebody
      objADOStream.position = 0    'Set the stream position to the start

      Set objfso = CreateObject("Scripting.FileSystemObject")
        If objfso.fileexists(strHDLocation) Then objfso.DeleteFile strHDLocation
      Set objfso = Nothing

      objADOStream.SaveToFile strHDLocation
      objADOStream.Close
      Set objADOStream = Nothing
    End If
    Set objXMLHTTP = Nothing
End Sub
```

```
Function multiReplace(ByVal expression As String, find As String, replacewith As String) As String
 'this function replaces multiple instances of a substing in a string with another string.
 ' it is helpful to change a arbitrary number of the same character to a single character
 ' such as converting all sequential spaces to a single space.

 Do While InStr(1, expression, find) > 0
  expression = Replace(expression, find, replacewith)
 Loop
 multiReplace = expression
End Function

Function replaceTags(theString As String, replaceTagsWith As String) As String
 ' this function replaces any HTML tags with the specified string.
 ' to remove tags, send a zero-length string as the second parameter

 Dim retval As String, dataon As Boolean, onechar As String, x As Long
 retval = ""
 dataon = True
 For x = 1 To Len(theString)
  onechar = mId(theString, x, 1)
  If onechar = "<" Then
   retval = retval & replaceTagsWith
   dataon = False
  ElseIf onechar = ">" Then
   dataon = True
  ElseIf dataon Then
     retval = retval & onechar
  End If

 Next

 replaceTags = retval

End Function

Function getText(theString As String) As String
Dim myPos As Long

myPos = InStr(pos, theHTML, theString)
 If myPos = 0 Then
  getText = ""
 Else
  getText = mId(theHTML, pos, myPos - pos)
  pos = myPos + Len(theString)
 End If


End Function

Function moveTo(theString As String, Optional ByVal theCount As Integer) As Boolean
   Dim x As Integer
   If theCount = 0 Then theCount = 1
   moveTo = True
   For x = 1 To theCount
    If Not singleMoveTo(theString) Then
       moveTo = False
```

```
      Exit Function
    End If
  Next
End Function

Private Function singleMoveTo(theString As String) As Boolean
 Dim myPos As Long
 myPos = InStr(pos, theHTML, theString)

 If myPos = 0 Then
  singleMoveTo = False
 Else
  singleMoveTo = True
  pos = myPos + Len(theString)
 End If

End Function
Function moveBackTo(theString As String) As Boolean
 Dim myPos As Long
 myPos = InStrRev(theHTML, theString, pos) + Len(theString)

  If myPos = Len(theString) Then
   moveBackTo = False
  Else
   pos = myPos
   moveBackTo = True
  End If

End Function


Function followLinkByText(thetext As String) As Boolean
 'clicks the first link that has the specified text
 Dim alink As Variant

  For Each alink In ie.document.Links
   If alink.innerHTML = thetext Then
     alink.Click
     waitForLoad
     followLinkByText = True
     Exit Function
   End If
  Next

End Function
Sub followLinkByHref(thetext As String)
 'clicks on the first links that has the specified URL

 Dim alink As Variant

  For Each alink In ie.document.Links
   If alink.href = thetext Then
     alink.Click
     waitForLoad
     Exit Sub
   End If
  Next
```

```
End Sub

Sub openpage(url As String)
 'opens the specified url in internet explorer
 ie.navigate url
 waitForLoad
 updateHTML
End Sub

Sub waitForLoad()
 'pauses the exection of the code until the webpage has loaded
 Do
  If Not ie.busy And ie.readystate = 4 Then
   Application.Wait (Now + TimeValue("00:00:02"))
    If Not ie.busy And ie.readystate = 4 Then
     Exit Do
    End If
  End If
  DoEvents
 Loop
 updateHTML

End Sub


Function readFile(path As String) As String
 ' returns the contents of a text file
 Open path For Input As #1
  readFile = Input(LOF(1), #1)
 Close #1
End Function
Sub savePage(Optional path As String)

 'saves the HTML of the current page to the specified path
 If path = "" Then
  Open ThisWorkbook.path & "\source.html" For Output As #1
 Else
  Open path For Output As #1
 End If
  Print #1, documentSource
 Close #1
End Sub

Private Sub Class_Initialize()
 initializeIE
End Sub

Private Sub Class_Terminate()
 terminateIE
End Sub
Public Function attach(URL_of_IE_Window As String) As Object
 ' used to attach to an already open IE window.  Requires the url of hte window
 Set ie = get_IE_handle(URL_of_IE_Window)
 updateHTML
End Function
```

```
Private Function get_IE_handle(URL_of_IE_Window As String) As Object
   Dim window As Object
   For Each window In CreateObject("Shell.application").Windows
     'Debug.Print window.LocationURL
     If window.LocationURL = URL_of_IE_Window Then Exit For
   Next
   Set get_IE_handle = window
End Function

Public Sub execScript(javascript As String)
   ie.document.parentwindow.execScript (javascript)
   waitForLoad
End Sub
Public Sub importPage(newSheetName As String, Optional wb As Workbook)
 'this procedure saves a local copy of a web page that the agent
 'is viewing and uses excel's web query functionality to import
 'the data to a worksheet.

 Dim s As Worksheet

 If TypeName(wb) = "Nothing" Then
   Set wb = ThisWorkbook
 End If

 savePage ThisWorkbook.path & "\localWebPageAgentFile.html"
 Set s = wb.Worksheets.add
 s.name = newSheetName

    With s.QueryTables.add(Connection:= _
     "URL;file:///" & Replace(ThisWorkbook.path, "\", "/") & "/localWebPageAgentFile.html", Destination:=s.Range("$A$1"))
     .name = "localWebPageAgentFile"
     .FieldNames = True
     .RowNumbers = False
     .FillAdjacentFormulas = False
     .PreserveFormatting = True
     .RefreshOnFileOpen = False
     .BackgroundQuery = True
     .RefreshStyle = xlInsertDeleteCells
     .SavePassword = False
     .SaveData = True
     .AdjustColumnWidth = True
     .RefreshPeriod = 0
     .WebSelectionType = xlEntirePage
     .WebFormatting = xlWebFormattingNone
     .WebPreFormattedTextToColumns = True
     .WebConsecutiveDelimitersAsOne = True
     .WebSingleBlockTextImport = False
     .WebDisableDateRecognition = False
     .WebDisableRedirections = False
     .Refresh BackgroundQuery:=False
   End With

   s.QueryTables(1).Delete

 Kill ThisWorkbook.path & "\localWebPageAgentFile.html"

End Sub
```