PinsRUs Solutions:

Pricing Tool & Quote Database

Brian Wade

ISYS 520

Executive Summary

Situation:

I happen to know the owner of a growing business called PinsRUs. PinsRUs designs and manufactures pins, medallions, and other similar products. The company is headquartered in Utah, but also recently acquired a factory in Kansas. PinsRUs does business all over the country, filling custom orders every day.

When PinsRUs receives a request for a quote (whether over the phone, in person, or through the company website) a company sales associate must perform the necessary calculations to determine what it will cost the company to fill the order, what price the customer should be charged to earn an acceptable profit margin, etc. The pricing model involves several variables, which differ with each custom order. The pricing also depends heavily on the cost of the metal that will be used in manufacturing, which can fluctuate significantly over time. On average, the owner of PinsRUs estimated that his salesmen probably spend 15 to 20 minutes going through all the calculations to provide a single quote to a customer.

Solution:

To help PinsRUs streamline and speed up the pricing process, I have utilized the knowledge and skills gained from our ISYS 520 class to develop a decision support system with VBA code in Excel. All user interaction with the program is through clean and simple user-forms. The program asks the user to input all necessary details about the order sought by the customer, accesses other information from a master data spreadsheet, and then performs all the necessary calculations almost instantaneously. The program then provides a cost and pricing breakdown with all the necessary information for PinsRUs and the potential customer. The quote breakdown is then saved to a searchable database, and can be accessed at a later time. This program was extremely well received by the owner of PinsRUs, and he estimates that it will (at the very least) cut the time required to prepare a quote by 50%.

Implementation Documentation:

1. Building the Code

1.1. Guiding Principles

Principle #1:	Explanation:
Automation	**Reasoning:** The purpose was to save as much time as possible for salesmen at PinsRUs. Any calculation that could be automated should have been and was automated. **Execution:** Users are only asked to input what is absolutely necessary. Everything else is pulled from a master table or calculated. The master tables worksheet is hidden in the workbook to avoid unintentional changes.

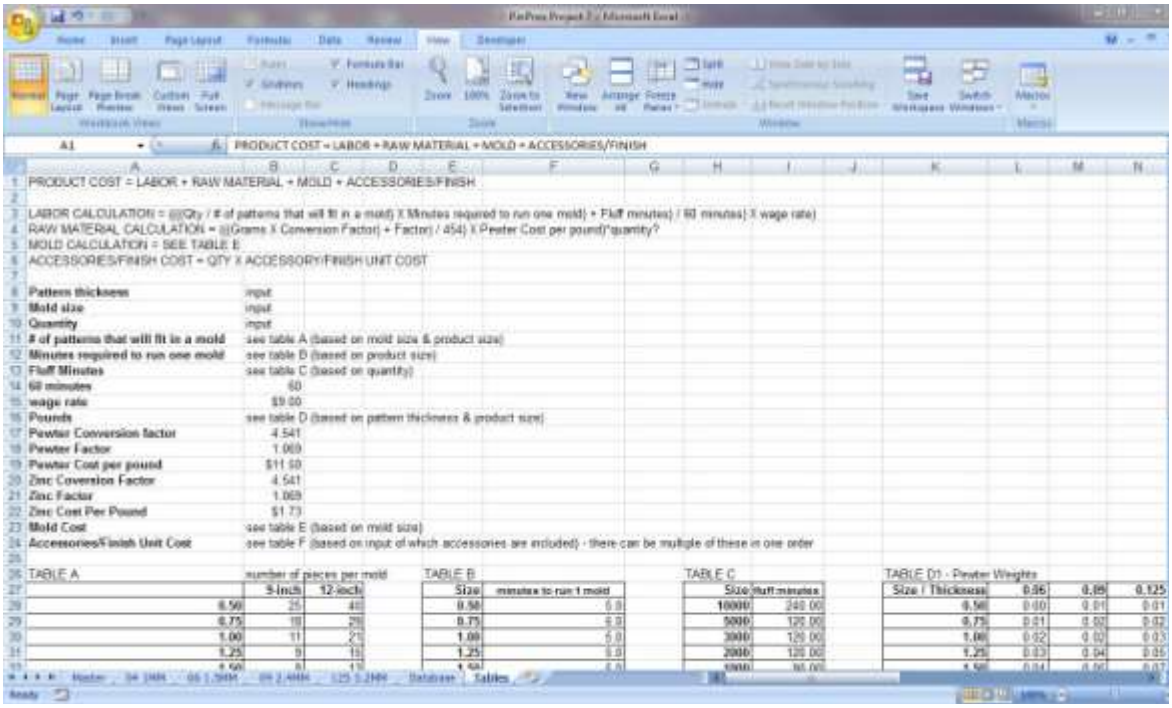


Figure 1.1 - a screen shot of the worksheet containing the formulas and master tables, wage rate, raw material costs, etc. The code pulls several pieces of information from this sheet, minimizing the load placed on the user.

Principle #2:	Explanation:
Usability	**Reasoning:** I wanted the program to be easy to learn and utilize. It needed to fit well with the needs of PinsRUs, and be designed with the user in mind. **Execution:** For example, the order in which information is inputted into the program reflects the logical flow of a conversation with a customer looking to place an order. This will minimize complications and speed PinsRUs employees down the learning curve once it is introduced as an everyday tool. Great usability will lead to seamless integration.

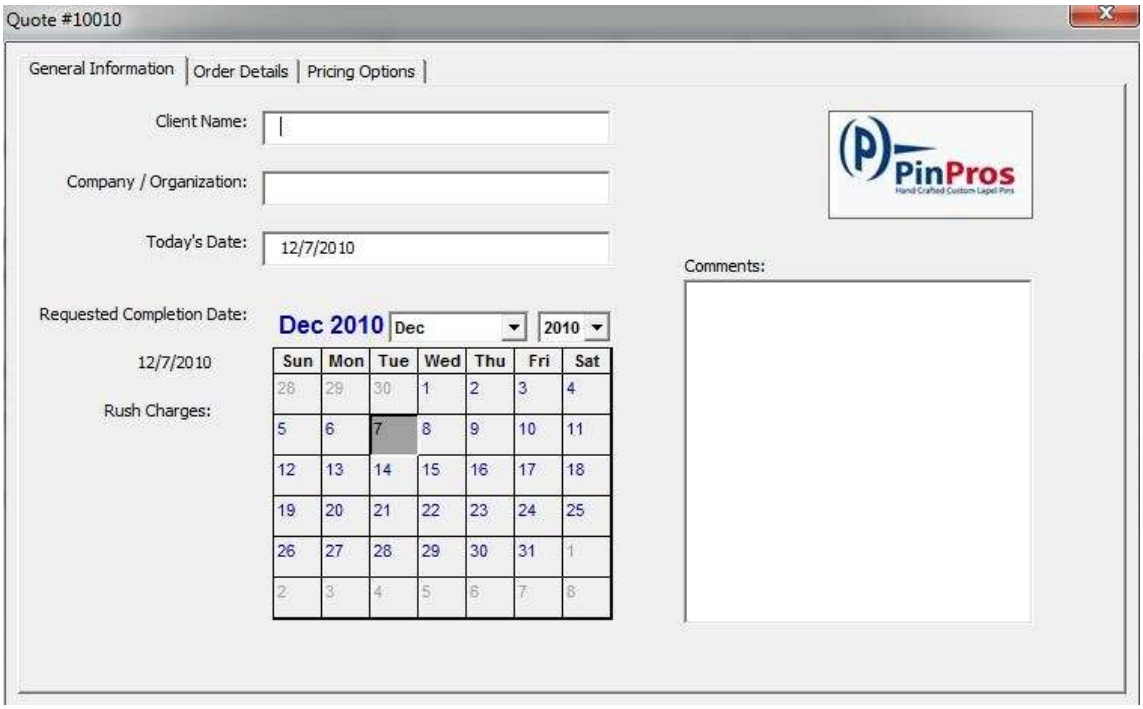


Figure 1.2 – In order to help salesmen intuitively use the program according to the natural flow of an interaction with a new potential customer, the first page that appears when a new quote is initialized asks for essential general information. The user can enter things such as the name of the client, the company or organization they represent, a requested completion date to ensure that the project is possible, and a comments section that allows for a brief description of the purpose of the order.

Principle #3:	Explanation:
Accuracy and Precision	**Reasoning:** Another intended benefit of the program is the minimization of error. Human error occurs frequently when performing calculations. While no program that requires user input can completely eliminate the possibility of mistakes, it can certainly help minimize them. The program is going to be used to price orders that are worth thousands and thousands of dollars. For example, a miscalculation of 10 cents in the unit cost could lead to a discrepancy of $1000 on a large order of 10000 pins. Thus, I wanted to make sure that the program would be error free. **Execution**: All calculations and formulas were double-checked when creating the code. I have also recommended to the owner of PinsRUs that he run several test quotes, checking the breakdown against his manual calculations, to ensure accuracy before introducing the program into daily use. In this way, hopefully any mistakes or bugs will be caught and worked out.

Figure 1.3 – By running several sample quotes and checking the resulting breakdown line-by-line against manual calculations, PinsRUs will be able to catch any possible errors in the code and fine tune the calculations. Once this has been done, they will have an efficient, precise, and accurate method of pricing that they can utilize with confidence.

Figure 1.4 – (see below) - Part of the long and complicated list of variables required for performing the final calculations. The rest wouldn't fit on the screen at one time

```
Dim accessoriesSelected As String
Dim finishesSelected As String
Dim orderqty As Integer
Dim piecesPerMold As Byte
Dim minutesPerMold As Single
Dim moldRuns As Integer
Dim moldCost As Single
Dim fluffMinutes As Integer
Dim poundsPerSquare As Single
Dim percentAreaUsed As Single
Dim poundsPerPin As Double
Dim accessoriesCost As Double
Dim finishCost As Double
Dim metalFactor As Double
Dim metalConversionFactor As Double
Dim metalUnitPrice As Single
Dim wageRate As Single
Dim moldLaborHours As Single
Dim moldLaborCost As Single
Dim rawMaterialsCost As Single
Dim afCost As Single
Dim finishesPerHour As Integer
Dim accessoriesPerHour As Integer
Dim finishLaborHours As Single
Dim accessoryLaborHours As Single
Dim afLaborHours As Single
Dim totalCost As Single
Dim profitMargin As Single
Dim basePrice As Single
Dim discountPercentage As Single
Dim discount As Single
Dim discountedPrice As Single
Dim totalLaborHours As Single
Dim rushCharge As Single
Dim finalPrice As Single
Dim unitPrice As Single
Dim totalProfit As Single
Dim rowSet As Integer
Dim columnSet As Integer
```

.

Principle #4:	Explanation:
Flexibility	**Reasoning:** The program had to be flexible enough to be changed without too much effort in the future. The business will expand, and new features and options will be added. Raw material input costs and wage rates will fluctuate regularly. The program must be able to account for these changes without being completely rewritten. **Execution:** This need for flexibility was kept in mind throughout the process. For example, the user forms incorporate a multi-page design. In this way, if more information or inputs are required down the road, another page can be added to the user form without having rebuild the form. If PinsRUs decides that they want to gather marketing data from customers when compiling a quote, for example, a new page could be easily added to incorporate several marketing survey questions. Also, several inputs are pulled by reference from the master tables sheet rather than defined within the code. If the wage rate or unit cost of pewter changes, for example, PinsRUs can simply change the master sheet and the program will use the new rates.

	A	B	C	D	E	F
11	**# of patterns that will fit in a mold**	see table A (based on mold size & product size)				
12	**Minutes required to run one mold**	see table B (based on product size)				
13	**Fluff Minutes**	see table C (based on quantity)				
14	**60 minutes**	60				
15	**wage rate**	$9.00				
16	**Pounds**	see table D (based on pattern thickness & product size)				
17	**Pewter Conversion factor**	4.541				
18	**Pewter Factor**	1.069				
19	**Pewter Cost per pound**	$11.50				
20	**Zinc Coversion Factor**	4.541				
21	**Zinc Factor**	1.069				
22	**Zinc Cost Per Pound**	$1.73				
23	**Mold Cost**	see table E (based on mold size)				
24	**Accessories/Finish Unit Cost**	see table F (based on input of which accessories are included)				

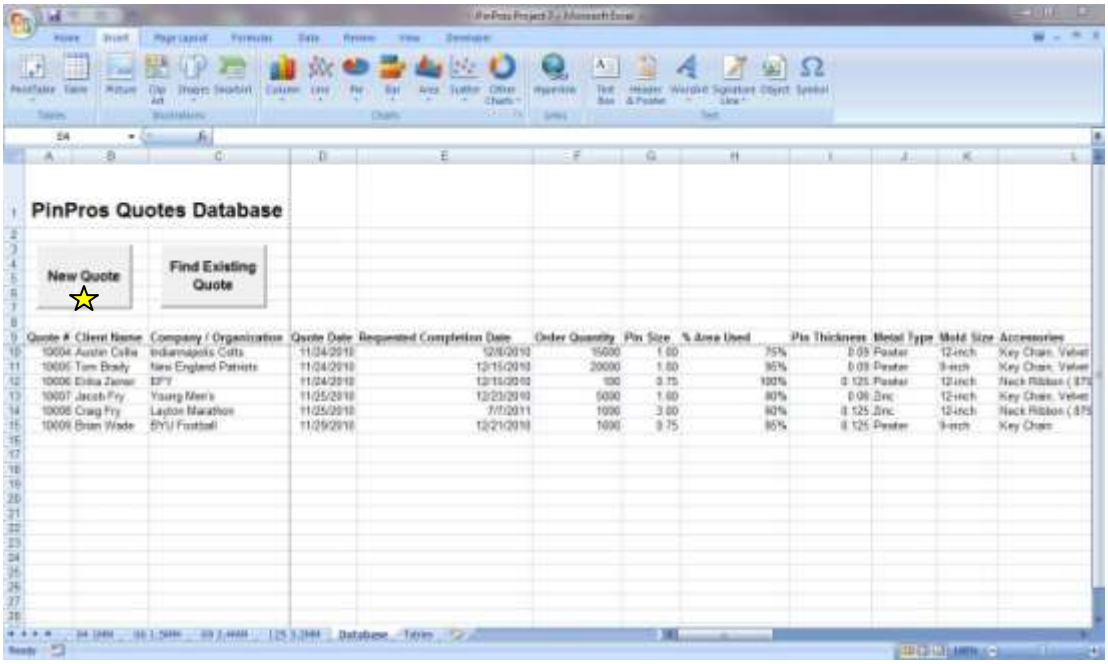
Figure 1.5 - part of the master tables worksheet, where the rates and/or locations of several necessary variable inputs are defined. This leads to increased flexibility in the future. For example, if PinsRUs raises their wage rate to $10 / hour, they can simply change the value in cell B15 and the program will automatically begin to use the new rate.

2. An Inside Look at the Resulting Program

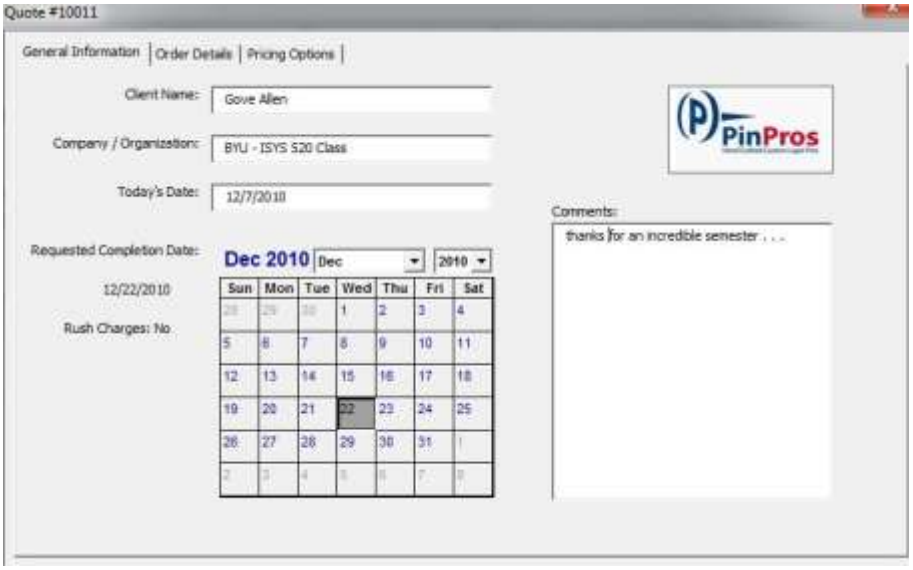
2.1. Stepping Through a Sample Quote

Following is a step-by-step breakdown of how the program works from the user's end. Where significant, explanations of the code-logic that drives the functionality are included:

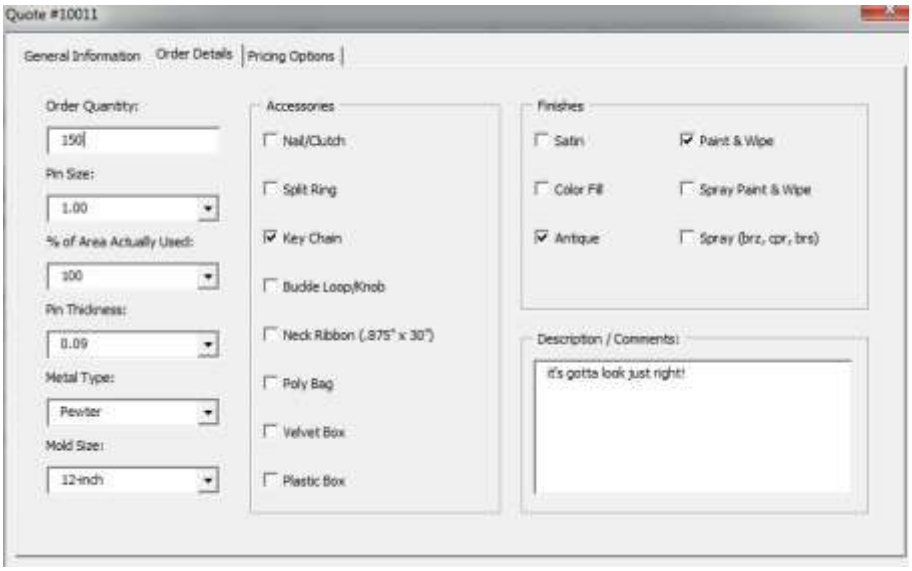
Step #1 – On the database worksheet, click on "New Quote". This command button is linked to a macro that assigns and enters the new quote number on a new row, and then opens the user form to begin a new quote.



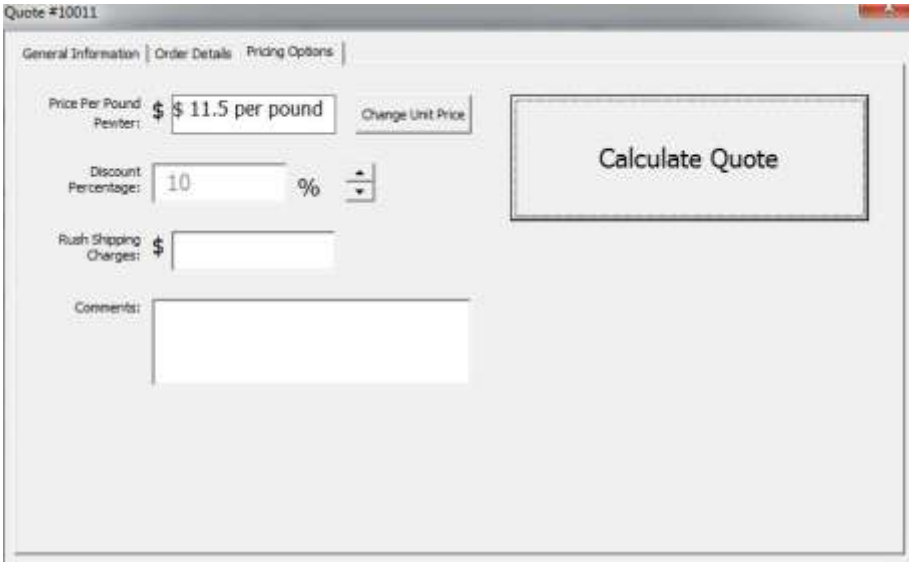
Step #2 – User must begin filling out the pages of the user form. Page #1 is general information.



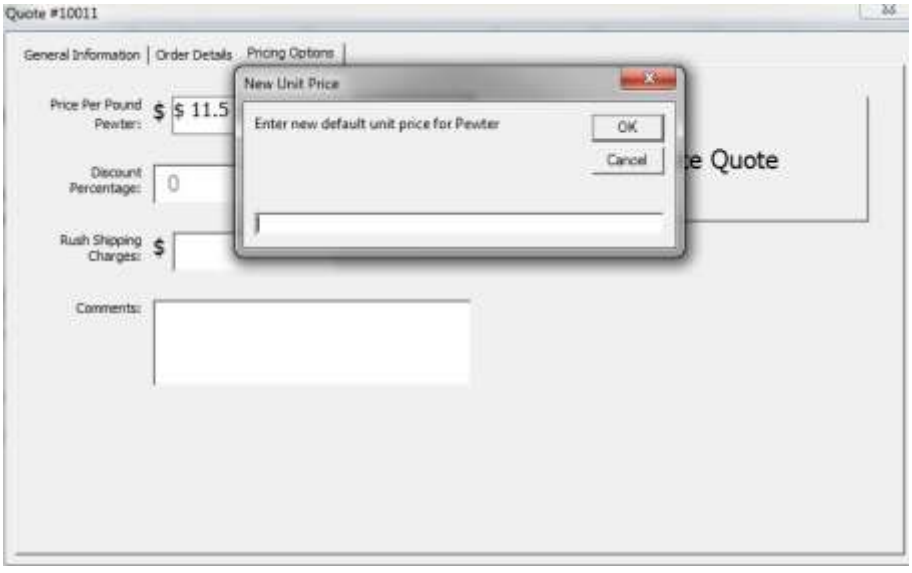
Step #3 – User must find out what the customer wants – Page #2 is order details. The user form has been populated with options from the master tables spreadsheet.



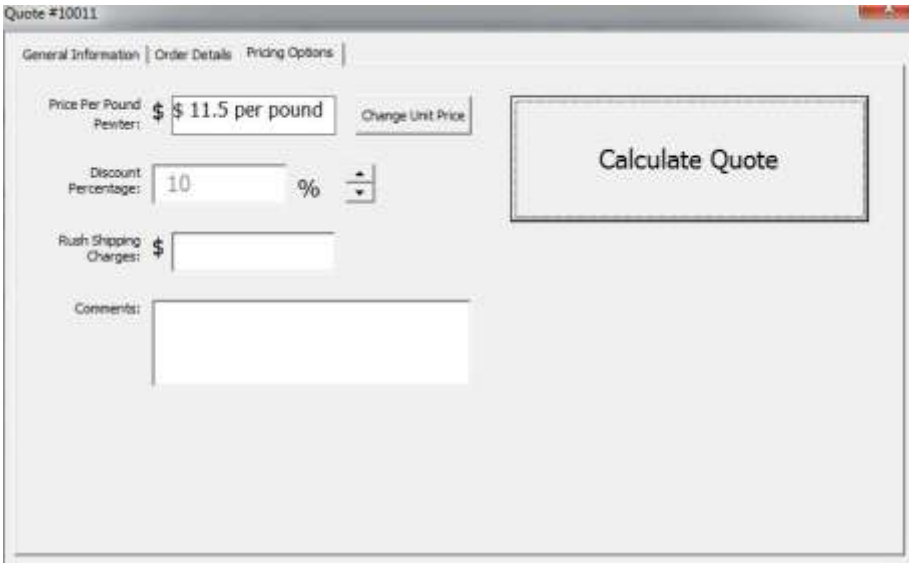
Step #4 – User must specify pricing options, such as discount percentage, rush shipping charges, etc.



Step #5 – If the metal unit price charged by the supplier has fluctuated since the last time a quote was issued, the user can click on “Change Unit Price” (see above). This will allow for a new unit price to be used for the current quote, and will also change the default unit price on the master spreadsheet.

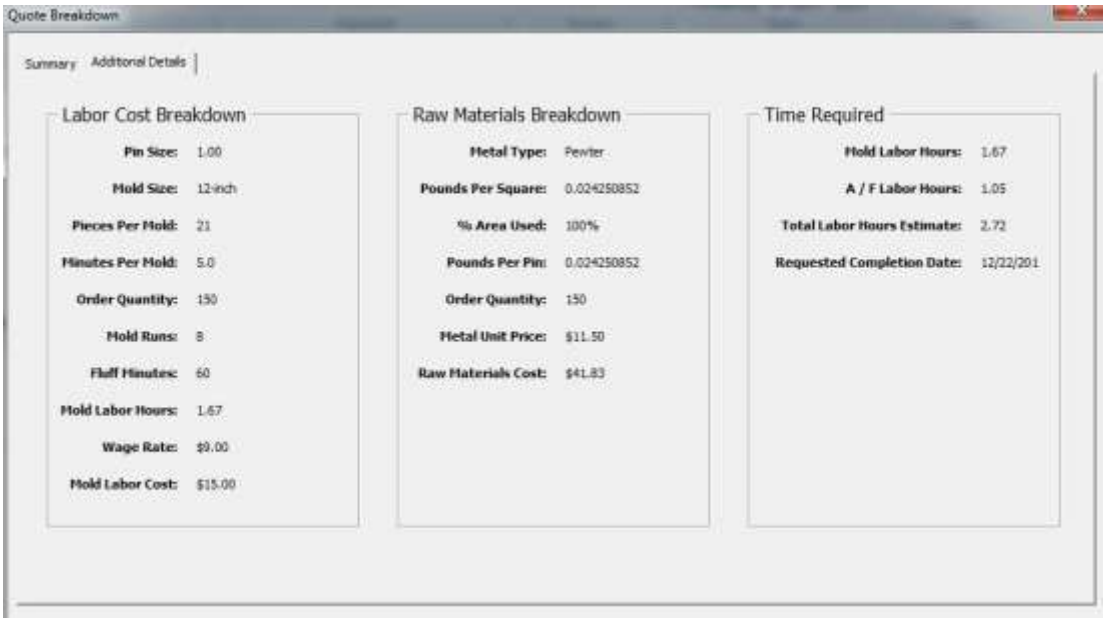


Step #6 – Hit “Calculate Quote”. The program will crunch all the numbers, write the results to the database row displaying the assigned quote number, close the current user form, and open a new user form displaying the quote breakdown.

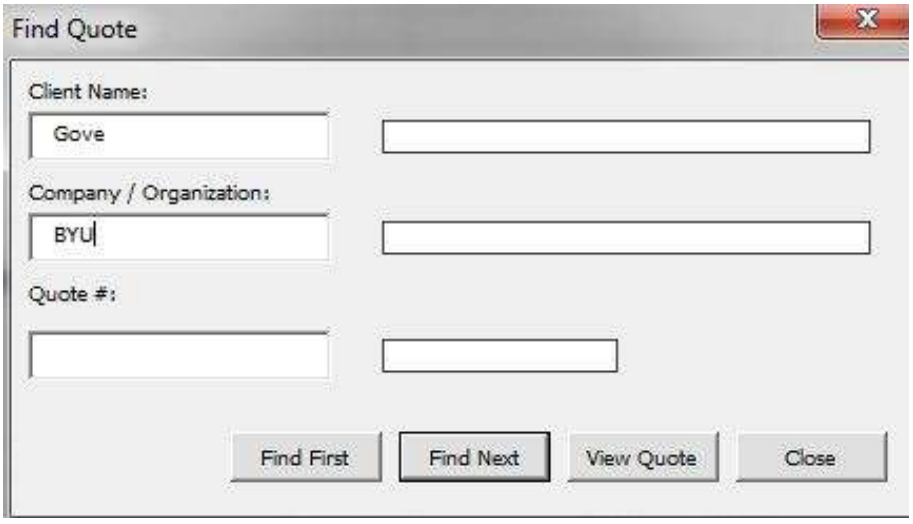


Quote Breakdown Page #1 displays a summary of the order and the cost breakdown, along with unit price, profit for PinsRUs, etc.:

Quote Breakdown Page #2 displays additional details about labor and raw materials, and provides an estimate of total man-hours required to complete the order:



Step #7 – If PinsRUs ever needs to refer back to a previous quote in order to verify something, the quote can easily be found and pulled up in the database using the find feature. This feature can be accessed by clicking on "Find Existing Quote", a command button which is placed right next to the "New Quote" button on the database worksheet (see previous explanation about starting a new quote).



2.2. Additional Features

The program code was designed with fail-safe measure so as to be as bullet-proof as possible. Many of these measures are incorporated when “Calculate Quote” is pressed. For example, if any essential information has not been entered on any page on the user form, the calculations will not execute but will bring the user back that spot and ask the user to input the missing information. If the user has accidentally selected a completion date that is prior to the quote date, an error message will be displayed.

```
'make sure have all necessary info:

    If Me.cboMetalType.ListIndex = -1 Then
        MsgBox "You must choose the metal type.", vbExclamation
        Me.MultiPage1.Value = 1
        Me.cboMetalType.SetFocus
        Exit Sub
    End If

    If cboPinSize.ListIndex = -1 Then
        MsgBox "You must choose the pin size.", vbExclamation
        Me.MultiPage1.Value = 1
        Me.cboPinSize.SetFocus
        Exit Sub
    End If

    If cboPinThickness.ListIndex = -1 Then
        MsgBox "You must choose the pin thickness.", vbExclamation
        Me.MultiPage1.Value = 1
        Me.cboPinThickness.SetFocus
        Exit Sub
    End If

    If cboMoldSize.ListIndex = -1 Then
        MsgBox "You must select the mold size.", vbExclamation
        Me.MultiPage1.Value = 1
        Me.cboMoldSize.SetFocus
        Exit Sub
    End If
```

Figure 2.1 - Examples of bullet-proofing in the code.

3. Learning and Overcoming Difficulties

Writing the code for this program took me dozens of hours. I'm sure that is probably the result of a few factors. First, I am a beginning programmer. I make stupid mistakes. There was one time where I spent a few hours trying to debug a part of the program, rewriting the logic in different ways, only to find out that it was a simple error that should have taken 10 seconds to spot. Sometimes I would need the same basic block of code multiple times, and so I would hastily copy and paste sections of code, forgetting to adjust a variable name here and an object reference there. It slowed me down in the long-run. This taught me how to be patient and meticulous in my programming.

Second, I had to go back and forth with PinsRUs on the phone trying to coordinate the project. I had to request additional data and information several times. However frustrating this might have been, these difficulties taught me the importance of information systems personnel. Understand the importance of information systems guys as the link between business and computer programmers – have to have interaction with the business guys, what they actually need, etc. that communication is key, must have someone who understands both sides – example – how much time would be wasted explaining a quote breakdown, how to calculate a profit margin / unit price, an understanding of the other things that might come up, etc.

There were many times where I needed to write a block of code or accomplish something that I had never done before. This taught me to utilize the help feature of VBA, to use resources on the internet, and to ask more experienced programmers for advice. For example, in dealing with trying to prevent the user from closing the user form by simply hitting the "x" button in the top right corner of the window, I found help on the internet. I was able to write the following code. Now, if the user tries to hit the "X" button to exit the quote, the program will display a message asking if the user is sure he or she wants to delete the current quote. If the answer is "no", the user form will resume without any loss of information.

```
Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
    Dim x As Byte
    If CloseMode = 0 Then
        Cancel = True
        x = MsgBox("are you sure you want to delete this quote?", vbYesNo)
        If x = 6 Then
            Cancel = False
            Unload Me
        For Each cell In Worksheets("database").Range(Range("a10"), Range("a9").End(xlDown).Cells)
            If cell.Text = quotenumber Then
                currentRow = cell.Row
            Exit For
            End If
        Next
        Worksheets("database").Cells(currentRow, 1).EntireRow.Delete
        End If
    End If
End Sub
```

I also struggled with preventing myself from expanding the scope of the project too much. I remember talking about this in ISYS 201, and reading that it is actually a major problem for many companies. It is extremely important to not let a project's scope expand so far that all semblance of simplicity and the originally desired functionality is lost. Programming can be as much about deciding what not to do as what to do. In writing the code, I had several new ideas that I thought would be really cool and I wanted to add additional features. However, those features were unnecessary to PinsRUs and they wouldn't be used. Luckily, PinsRUs was honest with me and kept me on target, letting me know exactly what they needed and what they didn't need. As a result, my final program may seem simpler than I would have liked, but it is also more streamlined, practical and useful than it would have been otherwise. I will remember that lesson for the future.

In conclusion, I really enjoyed my project. I learned a lot about programming, and plan to continue to use and develop my skills in the future. I talked with the owner of PinsRUs, and presented my project to him, and he really liked it. He is planning on having his salesmen use it, and is looking forward to saving time and money, even thousands of dollars each year.