Jeff Woodland

ISYS 520

12/8/2010

VBA Final Project: Personal Budget

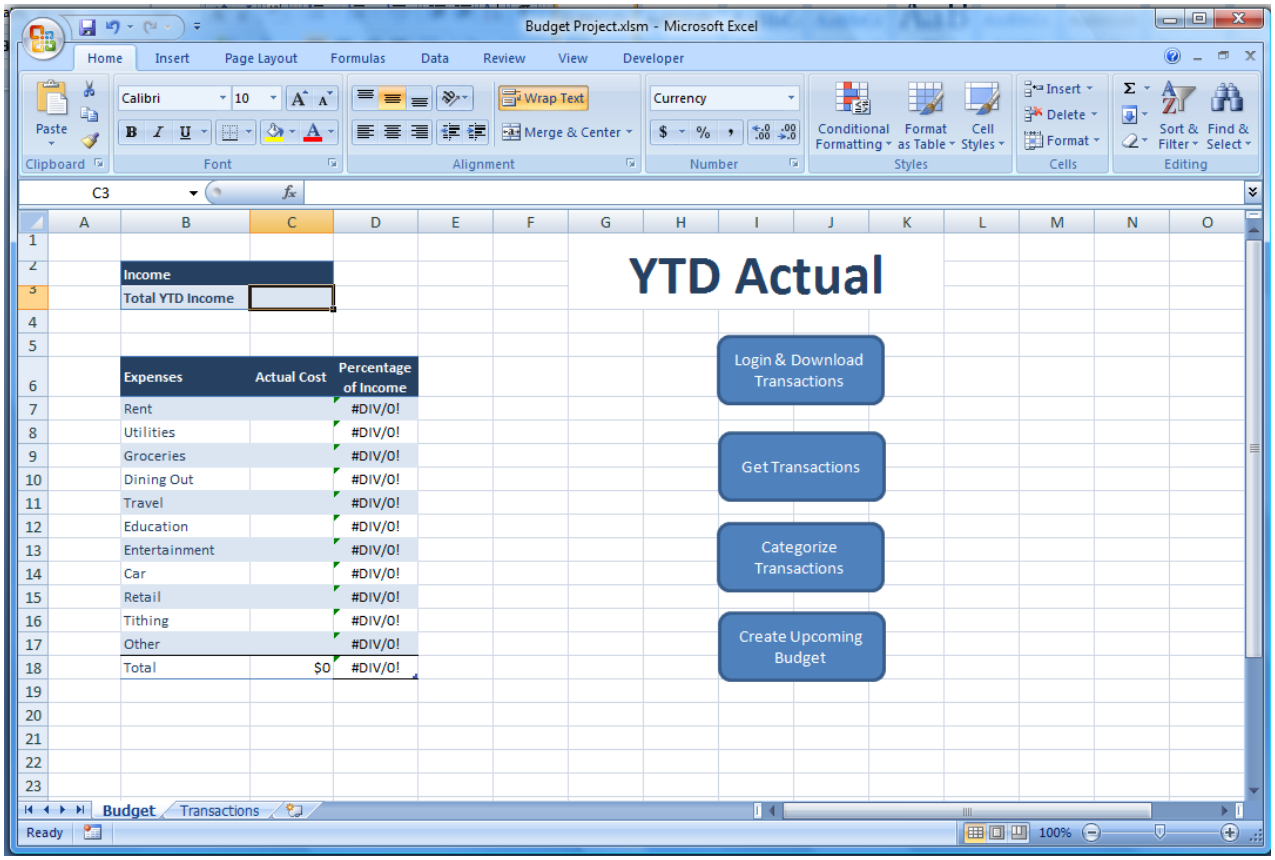
Executive Summary

Over the past few years I have tried to make and follow a family budget using an Excel spreadsheet. In each attempt, I have been foiled by the tedious nature of manually updating the budget. In any given month we could have anywhere between 50 and 100 different entries to enter into the spreadsheet. Predictably, these attempts to manually track my family's finances have been met with failure. Automating this process makes tracking finances much easier and allows more time to analyze expenditures and determine areas of improvement.

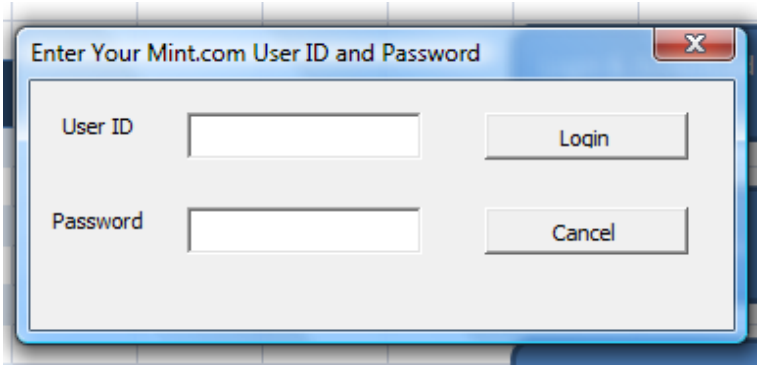
This spreadsheet greatly reduces the data entry portion of personal financial tracking by automating the process. The user enters login information for mint.com through a user form and the spreadsheet imports all transactions from the website. The spreadsheet then filters all transactions and compiles a table of all income and expenses from the data. For written cheques and other uncategorized transactions, the user has the ability to classify those transactions or leave them as "other" expenses. Finally, the user can create a budget for any given period of time to see how actual spending compares to the budgeted amounts.

Implementation

Upon opening the spreadsheet for the first time the user sees a blank budget table and some buttons in the Budget sheet and a blank worksheet called "Transactions."



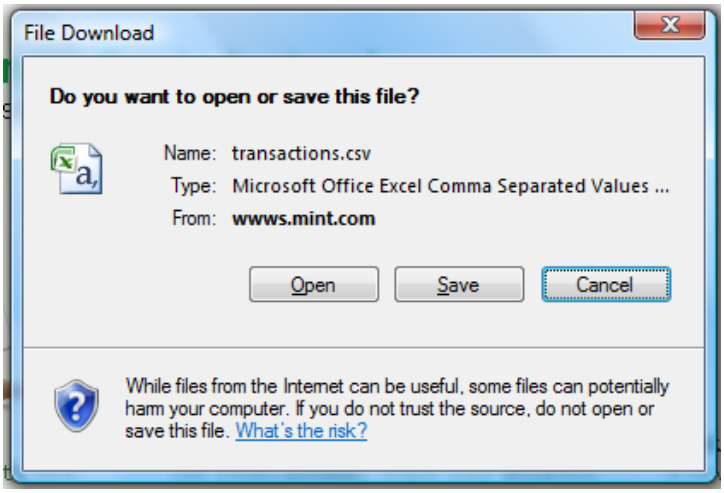
To begin using the spreadsheet, the user clicks the “Login & Download Transactions” and is prompted for a Mint.com user ID and password.



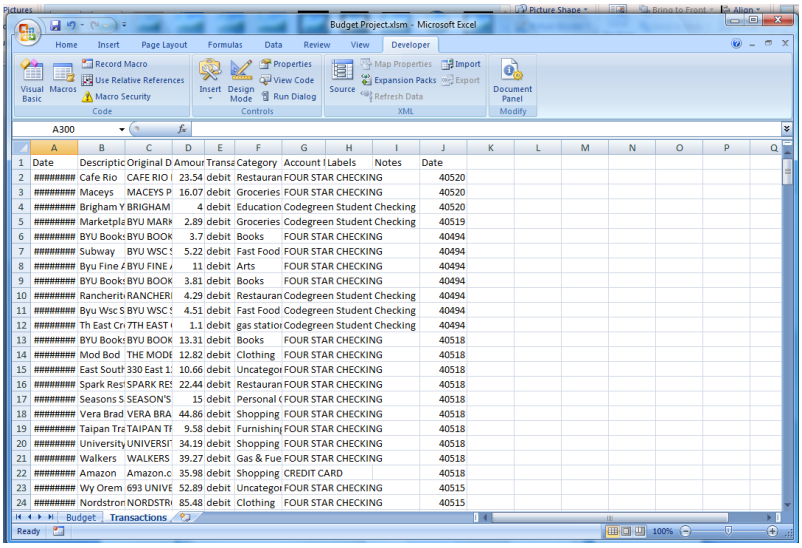
Mint.com requires an email address as the login ID so I decided to hard code my email provider directly into the VBA code to make login slightly easier.

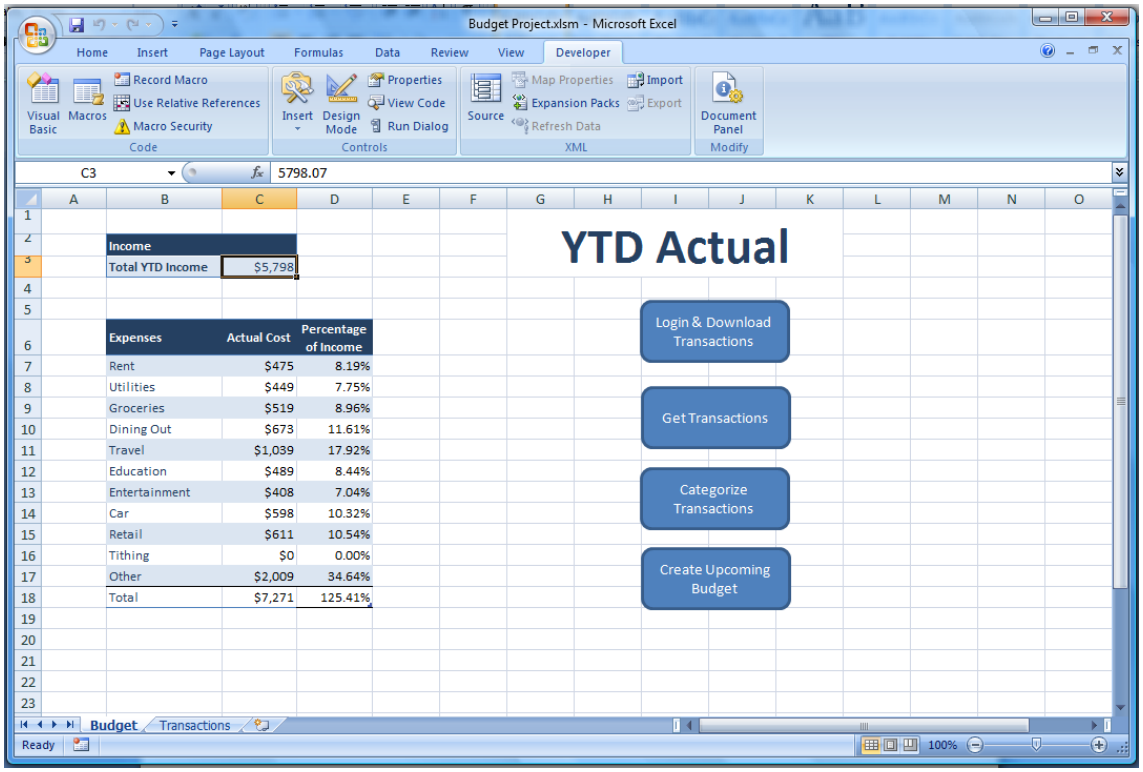
```
Sub Login()
agent1.visible = True

agent1.openpage "https://wwws.mint.com/login.event"

agent1.explorer.document.all("form-login-username").Value = frmLogin.txtuserid.Value & "@gmail.com"
agent1.explorer.document.all("form-login-password").Value = frmLogin.txtpassword.Value
agent1.explorer.document.all("form-login").submit
agent1.waitForLoad

getTransaction

End Sub
```

After logging in, Internet Explorer brings up a browser prompting the user to open a csv file in excel with all of the user's transactions compiled by Mint.com.

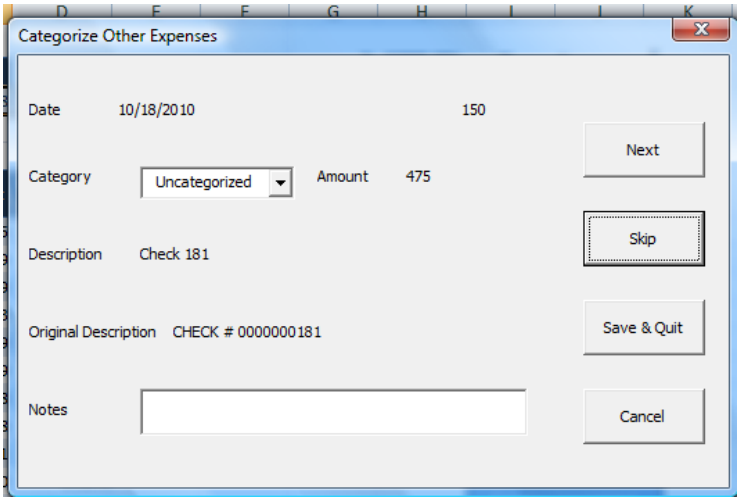


After some consultation with Dr. Allen, it was decided that the most effective way to interact with this window is to let the user click "Open" or "Save" to download and open the csv file and then click "Get Transactions" on the Budget sheet of the spreadsheet. After clicking "Get Transactions" on the spreadsheet the user now has all the transaction information on the Transactions sheet and the csv file has been closed. Also, the tables on the Budget sheet are filled in with the income and expense totals from each transaction.

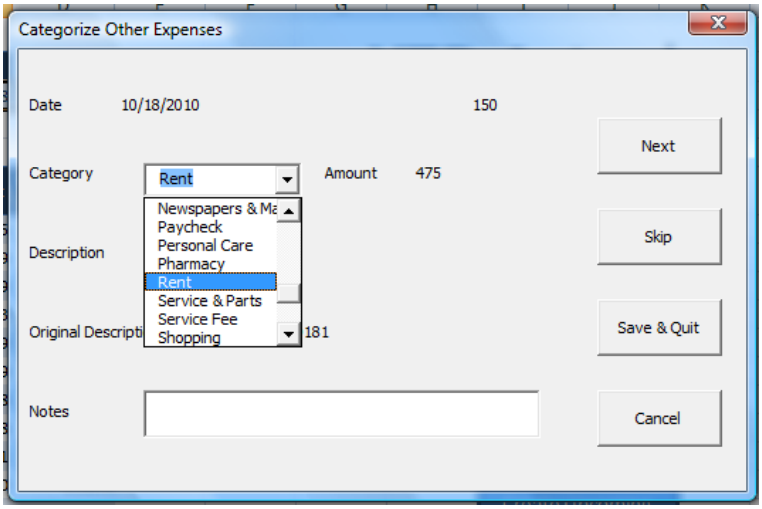




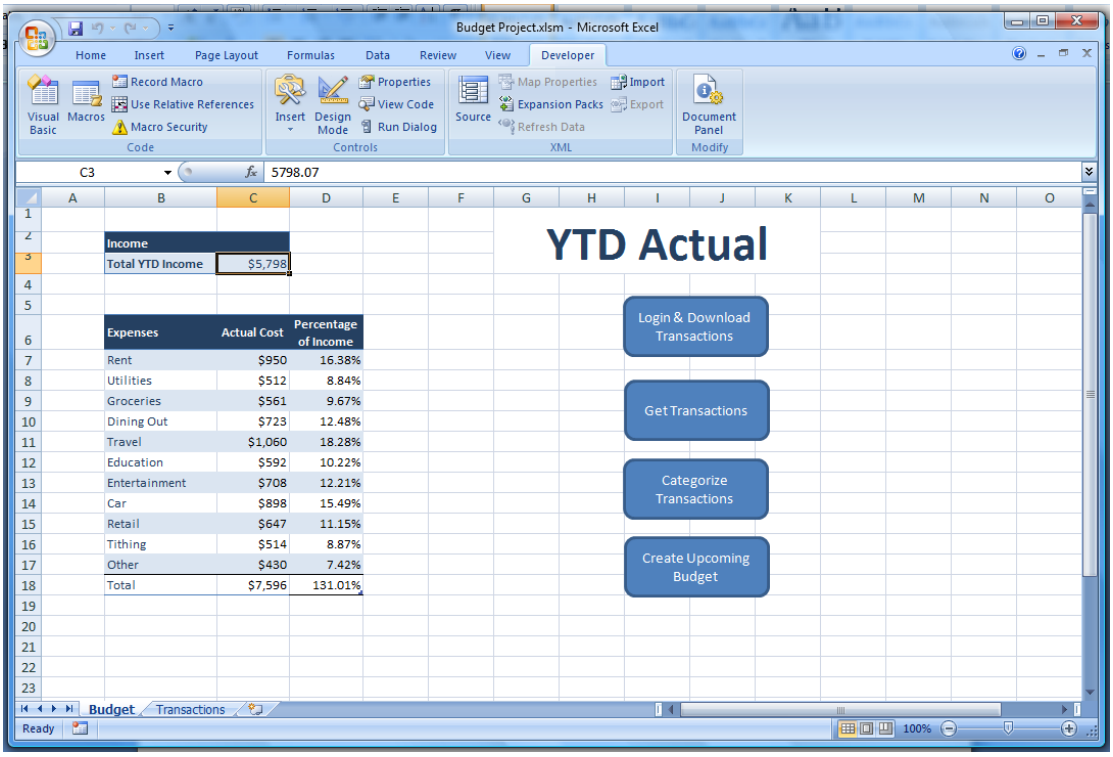
Even though the bulk of the transactions are automatically categorized by Mint.com, some transactions are in need of classification. Hence, the entry for "Other" expenses is significantly higher than the other classifications. The user can classify these expenses by clicking "Categorize Transactions." Clicking this accesses a form that allows the user to update only the category and add a note about the transaction.



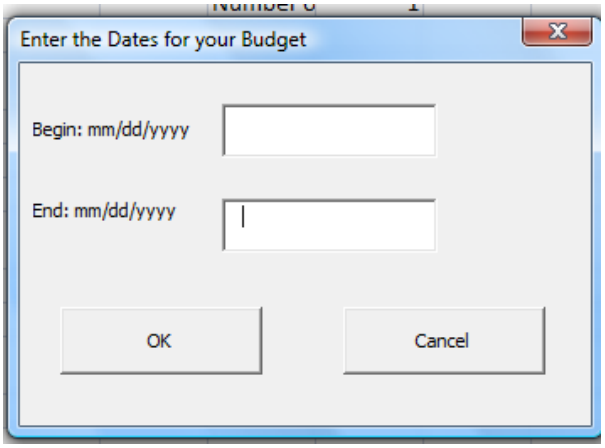
To make sure the subprocedure that filters the transactions into the correct expense categories works properly, the user can only select a category from a preexisting list. The user can save any changes made by clicking “Next” and moving on to the next uncategorized transaction, skip to the next transaction, save the current transaction and close the form, or close the form without saving the visible transaction.



After the user finishes classifying transactions, the budget automatically updates itself.

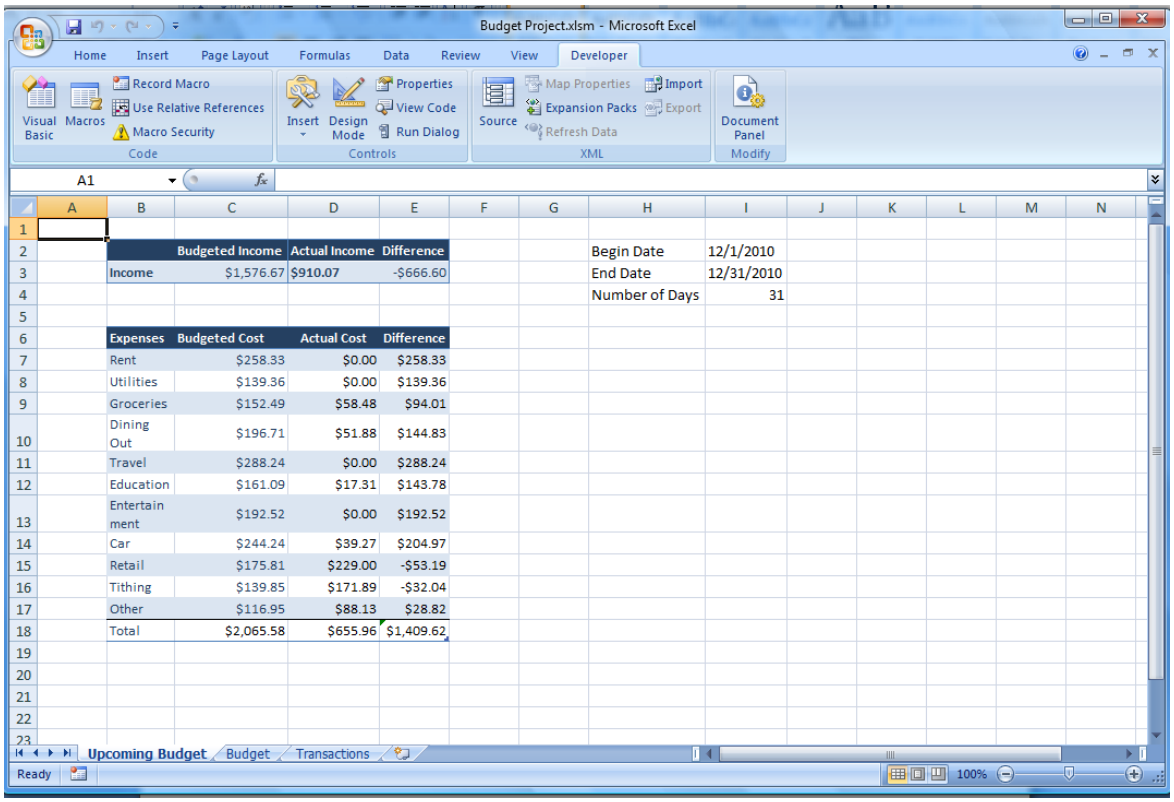


Once the user feels that the transactions reflect more accurately how her income is being put to use, the user can now use the spreadsheet to create a budget for any given length of time, past, present, or future. By clicking on "Create Upcoming Budget" the user is given a form to enter in the dates she would like to track.



The user is allowed to enter any date she likes. If the user were to leave a field blank the subprocedure will automatically ensure a date is given. If the beginning date is left blank, the procedure returns a year ago from the current date. If the end date is left blank, the end date used is the current date. The user is also instructed to resubmit the dates if she accidentally entered the dates in the wrong locations.

After entering the desired dates into the user form, the spreadsheet sorts through the transactions and enters all the entries that occurred during the given dates into the upcoming budget. To calculate the projected budget for the time period, the spreadsheet measures the average amount of each expense per day based on all transactions and allocates that amount based on the number of days in the time period. For example, if the user averaged two dollars a day on groceries, $60 would be allocated towards groceries for any given month.



Initially, I intended to create another user form to give the user more direct control over the creation of the projected budget in the upcoming budget sheet. However, as the project developed, it became apparent to me that it would be more advantageous for the user to start with an applied average of each cost category and manually increase or decrease the budgeted cost than to make the user come up with decisions from the outset.

Conclusion

This budgeting spreadsheet allows the user to easily get a grasp on his or her finances. Since nearly the entire process is automated, the tedious nature of personal financial management is gone. All the user needs to do is log on to mint.com through the provided user interface, download the transaction data and let the spreadsheet take care of the sorting. Through the use of this spreadsheet, personal financial management becomes much easier to manage.