David Lowe

ISYS 520 Sec. 01

Executive Summary

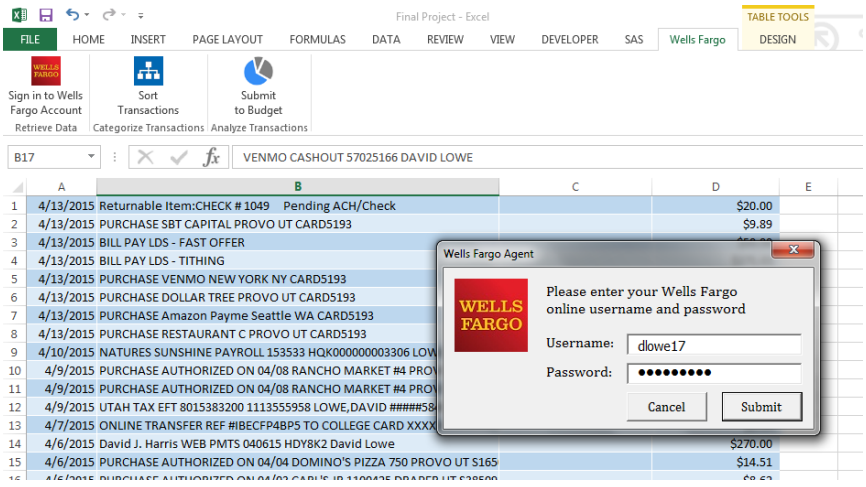
Account withdrawals and deposits, whether they be personal or business, are tracked by banks. If withdrawals are made with debit or credit cards those transactions are listed with the place where they were charged as well. Although most banks give information on where the card was used and how much was charged, many do not categorize the expenditure into personal or business budgets. Sorting through these transactions can be tedious and time consuming, especially if it needs to be done weekly or monthly. This program is designed to automate the majority of the process of categorizing transactions taken from bank statements. With this organizational system small businesses and individuals can easily extract their transaction history from their online account, choose budgeting categories, automatically categorize and sort transactions, and show monthly and running totals in spending for each category and the amount of disposable funds. By using this system the user can be more aware of where their money is going and can find more efficient ways to allocate income.

Implementation

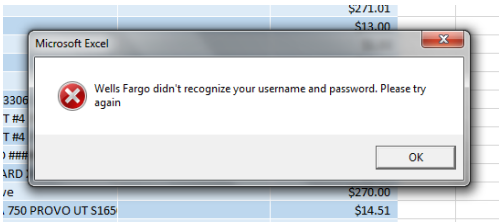
The excel workbook we are using is designed to not only work on short term transactions, but to be cumulative in its analysis. Therefore the more time this workbook is used, the more data is collected and our analyses become more and more accurate. The workbook is set up to organize, categorize, and analyze the data on a few different sheets. These sheets include, “Budgeting” (main analysis), “Monthly Totals” (all categorized transaction history for the month), “Running Totals” (all categorized transaction history up to the present), “Deposits” (all deposits made to the account up to the present), “Withdrawals” (all withdrawals made from the account up to the present), “Transactions” (all transactions up to the present), and “Imported Data” (temporarily used data taken from the web query).

Signing in/Data Collecting/Sorting:

The first step in this process was extracting the data from the online account. This required the use of the agent class variable that allows for ease of access into web browsers. This is connected to the “Wells Fargo” ribbon tab, “Sign in to Wells Fargo Account.” When clicked the web browser opens up the bank account web address (in this case the



Wells Fargo account entry page). At this step we need to enter the username and password of the account. This is done using a userform. Because we need to use the information put into the userform, this entire portion of web searching is done within the userform. The userform requires a valid username and password for a Wells Fargo account. The password text box uses the "PasswordChar" function to hide the characters entered.



Once submitted the form enters the given data into the Wells Fargo sign-in page and allows access to the account. With the agent class variable we access the link labeled "Account Activity," which provides the last 90 days of transactional history on the account. The agent class variable then is able to copy the entire page and download it into a sheet named "Imported Data." If the agent cannot access the Wells Fargo account then there will be no new information in the "Imported Data" worksheet and the macro will return an error box requesting re-entry of the username and password.

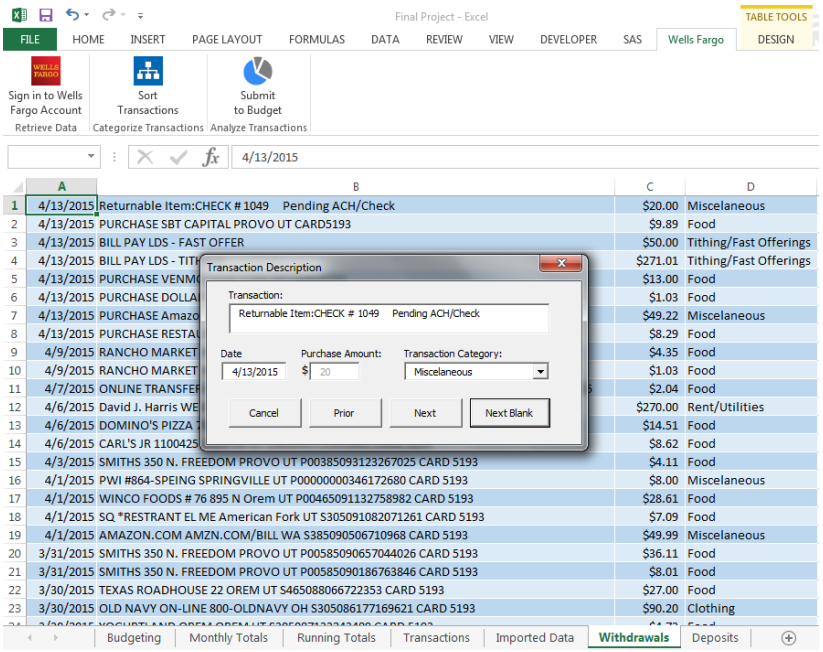
If the account is validated and the information was accessed then the "Imported Data" worksheet will contain the raw data file of the webpage, including a lot of superfluous information. To reduce the data into a useable form the macro uses a "do loop" to delete any row that does not have a date character in the first column. The only data on the page that provides dates in the first columns are the actual transactions made, which is what we are interested in. Once the desired data is the only thing left on the page the macro runs another "do loop" and a counter to see where the data in the "Imported Data" coincides with the first row of the "Transactions" worksheet. This row represents when the last update was made on the transaction history. The macro then deletes that transaction and all transactions before that transaction to leave "Imported Data" with only new data. Seeing as this workbook is designed to give monthly expenditures and income, it will be run at least once a month and would not surpass the 90-day transaction history limit. If the deadline were to be surpassed the macro would leave all of the transactions available for use from that 90-day period. In the case that there is no new information from the online transaction history then the macro stops, leaving the workbook just as it was.

With the new data found in the "Imported Data" worksheet we will be able to organize the data into "Transactions," "Deposits," and "Withdrawals." Conditional "do loops" insert each transaction row in chronological order from "Imported Data" into their respective worksheets depending on the column position of the dollar amount of the transaction. Now each of the worksheets, "Transactions," "Withdrawals," and "Deposits," have the updated transaction history from the online account.

Categorizing Transactions:

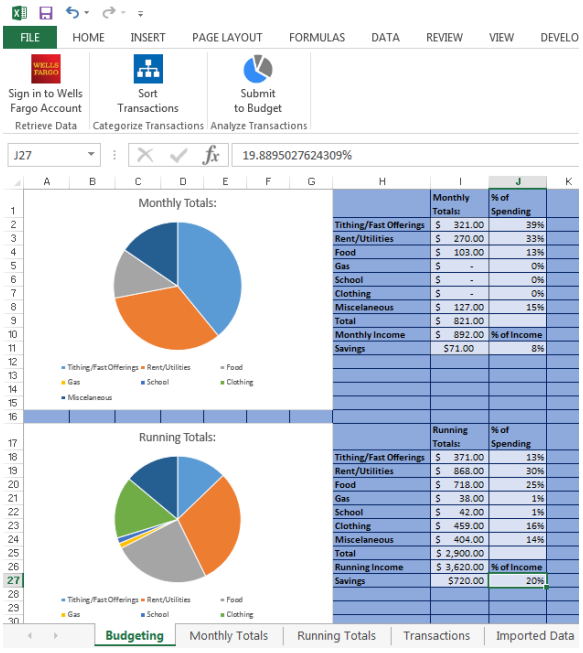
Now that each transaction has been sorted we need to categorize each withdrawal to form the budget. This can be done using the "Categorize Transactions" tab in the "Wells Fargo" ribbon. The first process is reducing each transaction description to only needed information by using an instr() function to take out superfluous information. In the "Withdrawals" worksheet we can use a userform that is programmed to help categorize transactions. This userform is programmed to look

for matching descriptions of transactions and when a category is already assigned to that description it will automatically fill in all similar transactions. When no matches are found the userform allows you to pick a category to place the transaction in. After each transaction that is categorized it checks to see if any matches are found for that transaction in any other uncategorized data and will fill in any matches automatically. This makes the process go much faster, especially when the program has been used over time and the majority of transactions are repeated at one time or another. The userform is designed to allow the user to move forward backwards and is defaulted to skip to the next transaction that has not been categorized. It also has error proofing built in to require every transaction have a defined category.



Creating Budgeting Statistics/Visuals:

Once all transactions are sorted we can use the "Analyze Transactions" tab in the "Wells Fargo" ribbon to insert each transaction into their respective categories in the "Monthly Totals" and "Running Totals" worksheets, where information can more easily be used. This allows the macro to find the totals for each category and compare them to the monthly and running total spending and income. With simple formulas created in the macro the "Budgeting" worksheet shows the percentage of spending for each category, along with a percentage of income being saved. This applies on both the monthly totals as well as the running totals. The running total percentages act as an average benchmark for the monthly percentages to be compared to. It also shows this graphically by creating pie charts that show what portion of the users funds are going where.



Learning & Conceptual Difficulties

The first problem I encountered was not knowing the full capability of the agent class variable. I worked by myself quite a bit before turning to my brother for help in understanding its functionality. Once I better understood it function and how to implement its capabilities it was very easy to take information off my account. The main problem I faced with this project is that it has a lot of different moving parts with the data. Probably due to inexperience I spent most of my time cleaning up the data to be useable. There seemed to be countless little and almost unperceivable issues with the loops I was creating to sort the data and transfer it to other worksheets. I received help mostly in debugging the code to find the little errors that were causing my code to run sometimes and other times not.

This project taught me the capabilities of the userform in much more depth than I had before. I also learned how to simplify several different problems in the data to one common solution. Much of my time was spent finding better ways to complete the task what I was doing. I began to see a lot of inefficiencies in my code that I could fix and it cut down processing time quite a bit. Some problems that seemed complex and above my ability to do, like the automatic categorization, turned out to be some of the easiest fixes with VBA. I also learned how to recognize and create protection against errors in my workbooks and code. This was an invaluable skill to learn because it made the last half of my project much more productive and efficient.

Once the data had finally been reduced and sorted how I wanted I began to think constantly about things that I could do to make relevant statistics for my bank account. I implemented a few with the running total average and the percent of savings, but there are many different ways to make this project even more useful. Due to time constraints I didn't apply all of them but this is definitely a project that I will continue to refine in the future. I didn't feel like I gave up on any ideas, rather I am brainstorming how to make them work. I am very aware of future questions I will want to answer with this data.

Outside assistance

While much of my code I was able to reason through myself, there were definitely parts where I needed some guidance. Because I was so engrossed in the problem, I often needed my brother or my cousin's help to recognize little issues with my loops and conditional statements. The internet was a huge resource for me to replicate processes and fit them to the needs of my project. I often found syntax and simple solutions on the internet that made my project run easier and more efficiently. The "record macro" tool in excel also helped me create a few charts and learn how VBA accounts for many of the simple tasks in excel. The help I received I feel was very beneficial to me and I feel that when I received help I was then capable of doing it myself. The learning curve is pretty steep still but it is good to know that I know how to resolve my problems in VBA.