Professor Allen
Michael Ryan
04/15/2014

VBA Project

Executive Summary:

Business Description

The accounting firm that this project was created for has many clients that need book keeping work done for them. The firm uses CertiflexDimensions (Certiflex) as their accounting software. Client journal entries can be uploaded from Excel into Certiflex if formatted correctly. The firm will often need to create these journal entries from bank statements and credit card statements.

Business Issue

Employees at the firm will copy client credit card information onto Excel and associate each entry with a Certiflex account. This is a very tedious and time-consuming process for employees, as client's card information may easily consist of thousands of entries a month. As employees enter information, they will make errors that go unnoticed and are hard to find.

Solution

I decided to automate this process as much as I could in order to save the company time/money as well as make the data entry process as error-free as possible. This project is split into four parts:

      (1) displaying instructions for the user to follow;

      (2) importing the data from AMEX online using the client's username and password;

      (3) reformatting the data, matching a Certiflex account number with each entry, and presenting a quality review interface where the user can easily check that the program matched the correct account with each entry make any necessary changes; and

      (4) formatting the data for Certiflex upload.

As most of the clients use American Express (AMEX) cards for business I decided to start with a macro for AMEX cards. This project has the capacity to expand into more uses and pull card information from other card companies.

Implementation Documentation:

Instructions –
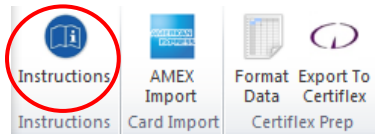
When the user clicks the "Instructions" button the project will open the "Instructions.pdf" file located at a predetermined location on the company's server using the "Shell" function. If the file is not there the macro will show an error. Using "On Error Goto…" the project will pull up a message box notifying the user that the instructions have been moved or deleted if the file is not found.
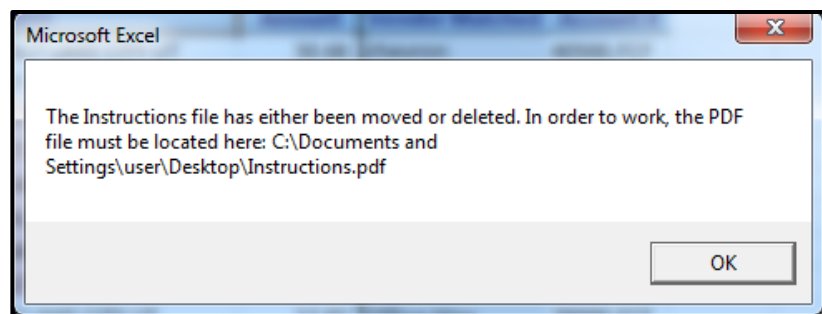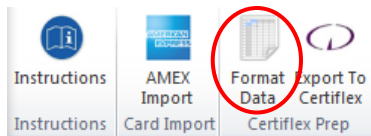
AMEX Import –
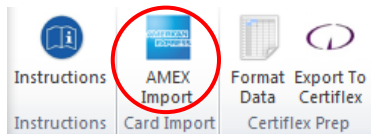
When the "Amex Import" button is clicked it will pull up the user form titled, "Please Input Information" (shown on the right). This form asks for the username and password (which is protected with the character "*") as well as a beginning and ending date range. The project uses the "Agent" Class Module to navigate the web. First, the program logs onto http://www.americanexpress.com using the inputted username and password. After some web navigation the project accesses the client's credit card statements. The program will input the beginning and ending date range from the userform into the

webpage. The webpage will pull up a statement with the desired date range. The project will click on a button on the webpage to import this statement from the internet into Excel. When the statement ("Summary.xls") has opened in Excel the project will copy the "Transaction Details" page from it and paste it into the project. After this is finished, the project will close Internet Explorer and close "Summary.xls" without saving. I've chosen to separate the data import sub procedure and the data formatting sub procedure so the user can download the data him/her self in the future if the webpage is changed and the macro no longer works. The bulk of the work (formatting) will still be automated.

Format Data –

When the "Format Data" button is clicked the project will search all the pages for a sheet called "Quality Review". If this sheet exists it will delete it. Next, the project will copy all the information from the "Transactions Detail" tab that was copied over from AMEX online. The project will place this copied data into a new sheet called "Quality Review". The program will associate this sheet with a publicly defined variable to make using this sheet easier throughout the project.

The project will clean up the data on this newly created "Quality Review" sheet by making all font consistent with the rest of the project (Size 11 Calibri); deleting the "Cardmember" column; adding the "Personal", "Type", "Ref", and "Vendor Matched" columns; and moving the "Account #" column. The project will also copy the header formatting to the other column headers as well as add a bold right boarder to separate the review columns ("Personal" and "Vendor Matched").

The project will remove the grey highlights on the card data and copy the formatting from the "Date" column and apply it to the non-formatted columns. Next, the project will give number formatting to each column according to its type. For example the "Ref" column is formatted as text, so any reference number starting with a zero (0452108001) will show the entire number and not cut off any leading zeros. All columns and rows will autofit and the review columns ("Personal" and "Vendor Matched") will be highlighted in light blue.

A title is added and formatted. A data validation list is added in the "Personal" column that will allow the user to choose either "Yes", or "-". If the user chooses "Yes" the column will highlight green. If the Vendor Matched shows "NotSure…" the row will highlight red. The program proceeds to delete any card payment entries – these entries are not needed.

The project will loop through each row in the data set and insert a new row at the end of each month. These rows are preserved for monthly totals and represent a card payable account. As the project loops through each row in the data set, it compares the month of the current row to the month of the previous row. If they are not equal (if the current row shows 09 and the previous row shows 08) the project will insert a row. The project will then fill the "Account #" column with a formula that will pull the account number from the Built-In vendor list (which is found on the "Vendor List" sheet) based off of the value in the "Vendor Matched" column. If the "Personal" column shows "Yes", then the formula will look up "Personal" on the vendor list. All the cells in the "Personal" column in the data range are filled with "-".

At this point the project will find all of these blank rows that were previously inputted and put the last day of the month in the "Date" column. The project will look at the date in the row above and take that month and year. If the month is April, June, September, or November the day will be 30. If the month above is February the project will pull the year into a function that determines if it is a leap year. If so it will be 29. If not it will be 28. If the month is any other month the date will be 31.

The project will fill in a string using the card number imbedded inside the string. The project will also place a SUM function in the "Amount" column that sums the month's card activity.

Next the project will fill the "Type" column depending on what type of Certiflex Entry each line is. Fortunately credit card entries only have two different entry types. Summary entries are type 5 and all other general entries are type 1. In the project these entry types are defined as public constants. The project loops through all rows in the data set and looks to see if the string is "AMEX" & accountNumber & "Payable". If so, the "Type" will be summary entries (type 5). All other entries will be general entries(type 1).

Next, the project will fill in the "Ref" column with a unique identifier for each entry in the data set row by row. The unique identifier for Certiflex must be ten digits and must be different from all other entries in the batch. The unique identifier consists of the last five digits of the card number (XXXXX), the month (mm), and a unique number (###). The last five digits of the card number is pulled from the publicly declared variable accountNumber, the two-digit month is pulled from the date, and the last five digits start at 001, increases 1 every row, and starts over for a new month. If the type of entry is 5 (Summary Entry) the last three digits of the unique identifier are "~JE".

Next, the project will go through the data set row by row and use the first word of the description (all the characters until the first " ") to find a match in the vendor list. The project will continue to search through the vendor list until all matches have been found. Each match will be entered into the combo box in the form "Matched Vendors" and the project will keep track of the number of matches found. The project will also go through all the vendors in the vendor list and input each one into the combo box in the form "All Vendors".

VENDOR LIST

	A	B
1	Vendors	Account #
2	4Imprint	70700.FCF
3	7-Eleven	40500.FCF
4	AAA Postal	70500.FCF
5	Aamco Transmissions	12105.FCF
6	Ace Hardware	70000.FCF
7	Advanced Vet	17000.FCF
8	AFM*AM Insurance	65100.FCF
9	AJ's Interior	12105.FCF
10	All Computer	60350.FCF
11	All for One Auto Parts	12105.FCF
12	All Year Store	12105.FCF
13	AM Auto Parts	12105.FCF
14	AM Family Insurance	65100.FCF
15	Amazon.com	72300.FCF
16	Amer. Fence	71000.FCF
17	Amex 48004 Payable	24059.FCF
18	Annual Membership Fee	60700.FCF
19	Antennax.com	12105.FCF
20	Arby's	60800.FCF

If there is only one match, the match will fill into the "Vendor Matched" column.

If there are multiple matches, the form "Matched Vendors" form is shown.

When the mouse hovers over the button "OK" the form will require a match. This forces the form to choose from the list before hitting the button "OK" so the text "Please select Vendor from list…" will not be inputted into the "Matched Vendor" column. Hovering over any other part of the form will make the form not require a match so the user may navigate out of the form. If the user clicks on the "OK" button the selection will fill into the "Vendor Matched" column. If the user clicks on the "Skip" button the "Vendor Matched" column will be filled with the value "NotSure…". If the user clicks on the "Vendor Not Shown" button the form will close and the form "All Vendors" is pulled up (see below for more detail on this form).

If the number of matches is zero, the form "All Vendors" is shown.

When the mouse hovers over the button "OK" the form will require a match. This forces the form to choose from the list before hitting the button "OK" so the text "Please select Vendor from list…" will not be inputted into the "Vendor Matched" column. Hovering over any other part of the form will make the form not require a match so the user may navigate out of the form. If the user clicks on the "OK" button the selection will fill into the "Vendor Matched" column. If the user clicks on the "Skip" button the "Vendor Matched" column will be filled with the value "NotSure…". If the user clicks on the "Make New Vendor" button the form will close and the form "New Vendor and Account Number" is pulled up (see below for more detail on this form).

When the form "New Vendor and Account Number" is pulled up the user will need to input a new vendor and the associated Certiflex account number. If the user clicks the "OK" button the values are inputted into the vendor list table in the "Vendor List" sheet. Also, the vendor is inputted into the "Vendor Matched" column. If the user clicks on the "Skip" button the "Vendor Matched" column will be filled with the value "NotSure...".

After a selection has been made the project will unload all forms.

If the user hits the red X on any of these user forms the remaining blank cells in the "Vendor Matched" column will fill with the value "NotSure...". This prevents the macro from showing an error and allows the project to finish running.

When all the cells in the "Vendor Matched" column have been filled a data validation list will be applied to the "Vendor Matched" column allowing the user to select from a list of vendors pulled from the vendor list table in the "Vendor List" sheet. This is done at the end of the project to allow for any new vendors created by the user to show up on these drop-down lists.

After the macro is finished running the user has the opportunity to perform a quality review. He or she can go through and check to see what vendor the system matched each entry with. He or she may choose to mark any entries as personal expenditures, which will be treated differently in the system (the account # will change if there is a "Yes" in the personal column). The green highlights on the personal expenditures allow the user to easily identify which ones have been personal. All expenditures with the "Vendor Matched" column that show "NotSure..." are marked red so the user may come back and change them to the proper account name, which will fill the correct account #.

Format Data –
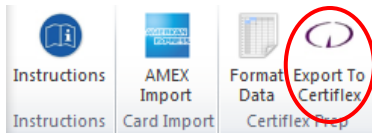
The user will click on the "Export To Certiflex" button after the user has verified that everything is correct, marked all personal expenditures, and corrected all "NotSure…" entries. The project will display a message box if the user has not cleared all the "NotSure…" entries.
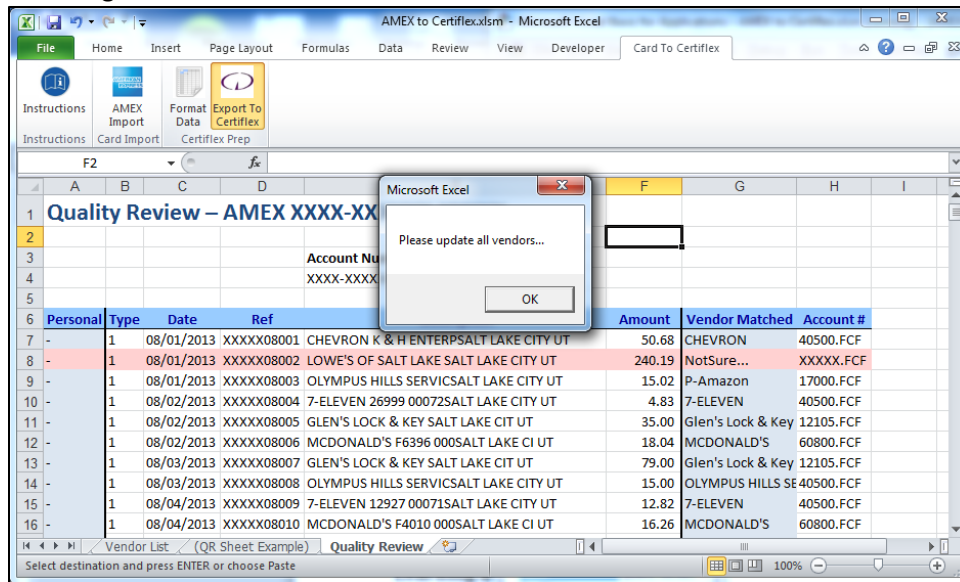
If all "NotSure…" entries are cleared the system will search the workbook for any sheets named "Upload To Certiflex". If any are found, the project will delete the sheet. The project will make a new "Upload To Certiflex" sheet. Next, the project will copy the "Quality Review" sheet and paste value the information into the new "Upload To Certiflex" sheet. The values from "Vendor Matched" are copied over top the "Description" values. The "Personal" and "Vendor Matched" columns are deleted, a new row is inserted between the "Discription" and "Amount" columns, and the "Account #" column is cut and copied in the newly inserted column. The cells above the table are also deleted. The table is now ready to import into Certiflex.
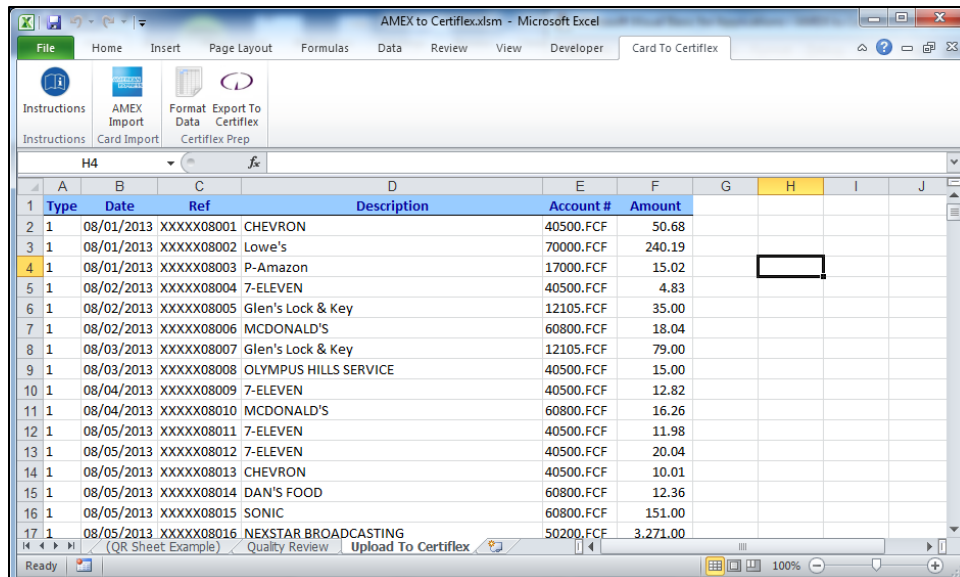
Learning and Conceptual Difficulties:

Using the "Agent" to navigate through Internet Explorer was difficult for me. I didn't understand how to find the right thing on the HTML. I also didn't understand that you could actually have VBA "click" on certain buttons. This was a great learning area for me.

I also had a difficult time conceptualizing the vendor list matching. My challenge was to make it as automated as possible. It was also hard to figure out how to get the project to return the correct result with all the forms behaving in the way I wanted them to. The user has the option to switch from form to form, so it was difficult to figure out how to get the project to pull the value from the right form. This part of the project took a lot of research online.

I also spent much time online learning various things about the user forms. I wanted them to behave in a certain way, but I changed my approach to cater to the limitations of the user forms. For example, the "Match Required" property of the combo box was a struggle to work with – sometimes I would need it on but sometimes having it on would mess things up, so I discovered the MouseMove option to make that property toggle on and off when I moved my mouse over various parts of the form.

Assistance:

Nathan (the TA) has given me help on the data import part of this project. I also spoke with some coworkers to form some big-picture strategies of how to present some of the Quality Review aspects.