

final  
project  
vba

*Mango*  
*properties*  
*expense*  
*report*

*1.13.11*

mba  
614  
pablo arias

## Executive Summary

Real Estate has always been a goal and a dream for me. After marriage, our first goal was to buy our own house and build enough equity to keep investing in houses. We were lucky to buy our first house at the right time and ended up selling in the perfect time. With the experience we had, we felt that we could continue to have success in the real estate market.

Almost 10 years later, my wife and I now have four rental properties that we have acquired through the years. All four properties are located within a 10 miles radius in the Salt Lake Valley area. Managing these four properties is a challenge. Real estate is risky and not for everybody. You need to learn how to deal with people and issues you never thought existed. We learned a lot from our own experience, and every time we learn something new, we move on and try not to commit the same mistake again.

One of the things we do every month is to drive around our houses and see if they are still standing. Another thing we do is to verify if rent has been paid. We also keep a close relationship with our renters because we like avoid surprises. Overall, we are glad to have the extra cash flow from the properties to help us with our own expenses. We make sure mortgages are paid on time and we invest often in the properties, fixing anything that is broken and reported by our renters.

One thing we need to improve is the book keeping aspect of our business. It is often the case that we don't keep track of how much we spend in repairs and maintenance. Our mortgages and a few other expenses are set up in an automatic withdrawal system. The intention of this project is to provide a way in which we can run a report in excel that will tell us how we are doing in our property related expenses. We want to be able to press a few buttons and see a report that will tells us how much we have spent in the last 12 months. Knowing what our income is, it would be nice if we could verify our expenses and see how much our profits are.

The project is protected by a personal login and password, which matches our Wells Fargo online login and password. The report works in the following way:

- I open the file in excel
- I pull the data from the Wells Fargo web page
- I process the data to keep only the information I want in my report
- The VBA code separates the data by month and adds the total expenses for each month
- The VBA sends the total expenses for each month to the summary page
- The VBA creates a graph to show the trend of expenditures for the last 12 months

## Implementation Documentation

My first step was to create an excel file with a table to document the findings (see Figure 1). This table would ultimately show the total expenditure amounts for each of the previous 12 months. Once I figured out a way to bring the information to this table, I would be able to create a graph to illustrate the findings (see figure 2).

Figure 1

Last 12 Months	Balance
Total Expenditures	\$ -

Figure 2



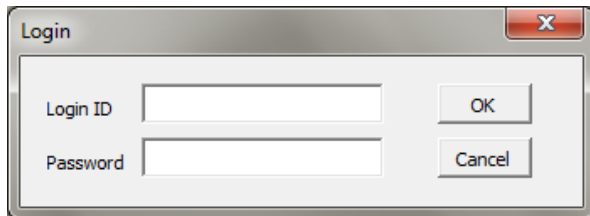
To pull the information from the web, I created a command button called “Get Data” (see Figure 3).

Figure 3



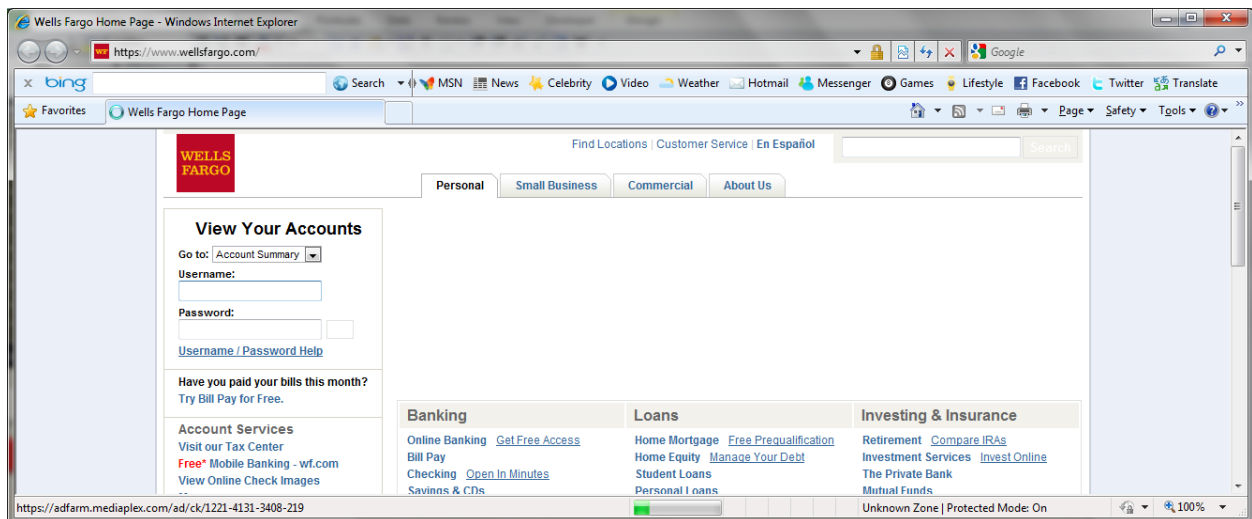
By pressing the button, the VBA code then activates a sub procedure called “loadForm” which brings up a user form called frmLogin (see Figure 4).

Figure 4

A login form titled "Login" with a close button (X) in the top right corner. It contains two input fields: "Login ID" and "Password". To the right of the "Login ID" field is an "OK" button, and to the right of the "Password" field is a "Cancel" button.

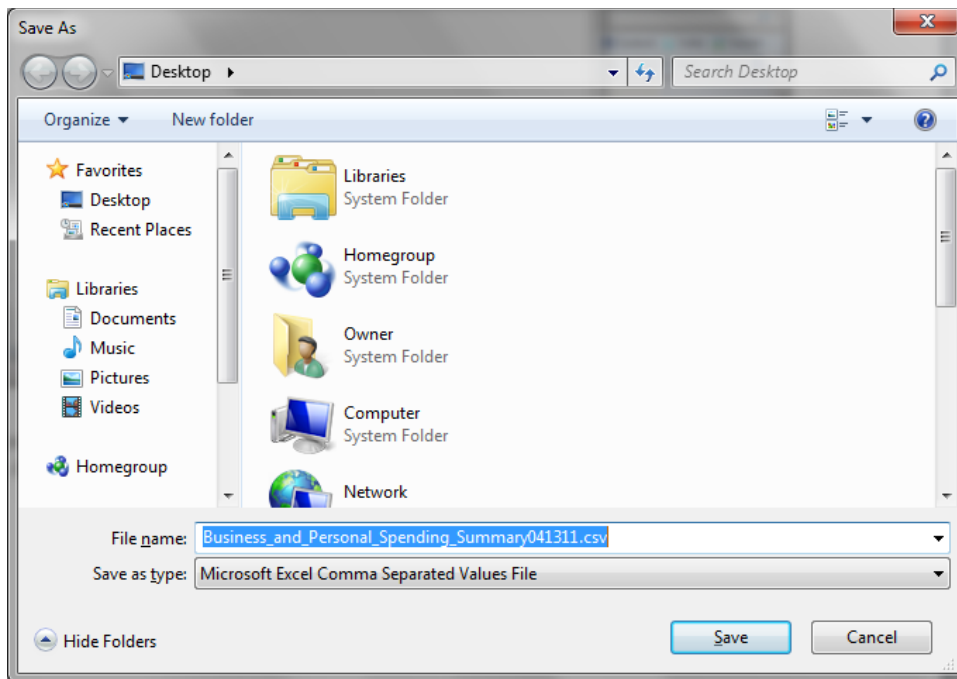
This user form prompts the user to enter the Wells Fargo login and password that will enable the macro to pull information from the web. If the user presses Cancel, the user form closes and nothing happens, if the user presses OK without entering any information, a message box appears prompting the user to enter a valid login and password information. If the user is successful in entering the correct login and password, excel opens its browser and enters the Wells Fargo page (see Figure 5).

Figure 5



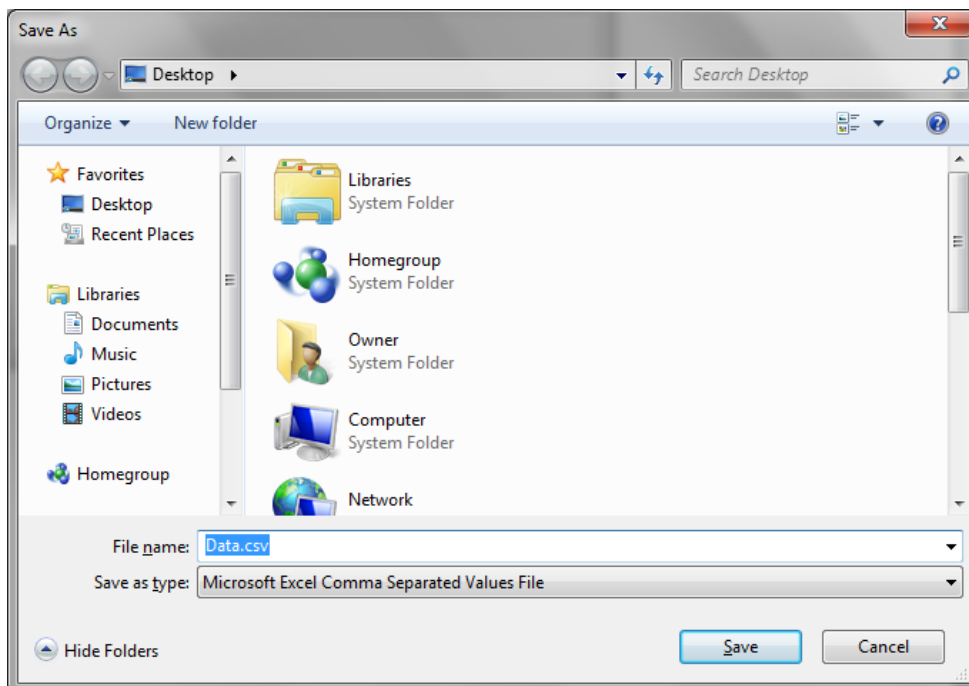
Excel then uses the information given in the login and password to enter the Wells Fargo site and retrieve information from the server. Excel then prompts the user to save a file called Business and Personal Spending Summary (see Figure 6).

Figure 6



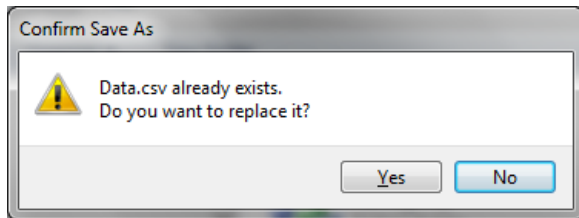
This is a critical part of the procedure since the VBA code will look at a specific area in your computer to pull the data into the file. The user then needs to make sure he is saving the file with the name Data, and the file must be save in the user's desktop, in the following way (see Figure 7):

Figure 7



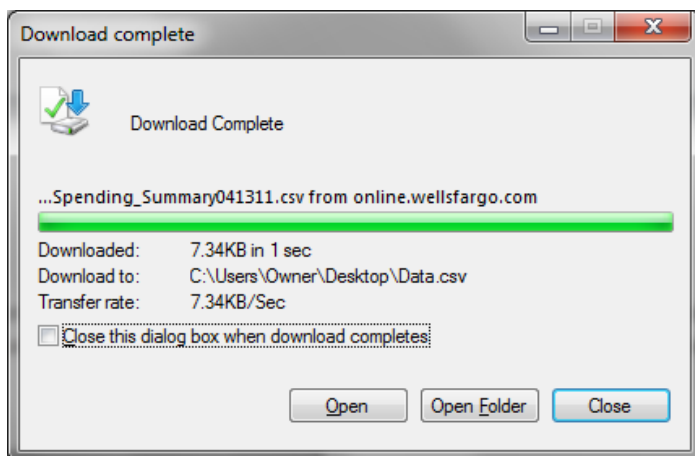
After saving, the computer will ask if the user wants to replace the file already in existence, which will happen only if the user has used this system already and saved this same file previously (see Figure 8).

Figure 8



For the purpose of using the most updated data, the user should say yes and save the data in the user's desktop. After saving, the computer will ask if the user wants to view the data (see Figure 9) and the user should select close.

Figure 9



After pulling the data from the Wells Fargo web page, the excel file is ready to process the data and arrange it in a way that will provide the desired total expenditure amounts for the each of the last 12 months. To process the data, the user needs to click on the button "Process Data" (see Figure 10) which will call a sub procedure called "ProcessData."

Figure 10



What this sub procedure does is to insert a new worksheet and label it "Raw Data" and then put in it all the data that has been downloaded. The data will include dates, the bank account, the type of transaction that was performed, and the amount spent. Since all I am interested on is the date, the

transaction type, and the amounts, the ProcessData sub procedure deletes the other columns and leaves on the worksheet “Raw Data” only the pertinent information. Once it does this, the sub procedure also selects the data and sorts it by date, from latest to newest dates. The Raw Data worksheet will have the following information (see Figure 11):

Figure 11

Posting Date	Description	Amount
4/13/2010	ONLINE TRANSFER REF #IBE82PKNWP TO MORTGAGE WEST VALLEY MOR	\$1,072.82
5/11/2010	ONLINE TRANSFER REF #BEFTX5BLV TO MORTGAGE MORTGAGE PAYMEN	\$1,072.82
5/17/2010	CHECK	\$542.62
5/18/2010	CHECK	\$385.00
6/7/2010	ONLINE TRANSFER REF #BEQK727NT TO BUSINESS MARKET RATE SAVINGS TO SAV	\$112.99
6/10/2010	ONLINE TRANSFER REF #BEMN5WHYB TO MORTGAGE 3438 S 3690 W WE	\$1,072.82
6/14/2010	CHECK	\$989.59
7/6/2010	CHECK	\$1,887.00
7/21/2010	CHECK	\$989.59
8/9/2010	CITIMORTGAGE INC LOAN PAYMT 080910 20045011523 MANGO PROPERTIES, LLC	\$1,286.76
8/13/2010	CHECK # 106	\$335.00
8/16/2010	CHECK CRD PURCHASE 08/13 THE HOME DEPOT 4403 SALT LAKE CIT UT 425909XXXXXX3734 2	\$32.67
9/3/2010	CHECK # 107	\$50.00
9/7/2010	CHECK # 109	\$100.00
9/7/2010	Rocky Mtn Power Payment 090310 00108 902460079640292 # 108	\$18.14
9/8/2010	CITIMORTGAGE INC LOAN PAYMT 090710 20045011523 MANGO PROPERTIES, LLC	\$1,286.76
9/9/2010	ONLINE TRANSFER REF #IBEJR9JLW TO MORTGAGE WEST VALLEY SEPT M	\$902.32
9/10/2010	CHECK # 110	\$2,185.00
9/15/2010	CHECK # 111	\$989.59
9/20/2010	Rocky Mtn Power Payment 091710 00112 902600092610316 # 112	\$9.69
9/22/2010	CHECK # 113	\$50.10
10/1/2010	CHECK # 114	\$22.28
10/5/2010	ONLINE TRANSFER REF #BEJRMSMGZ TO MORTGAGE WV OCT MORTGAG	\$902.32
10/7/2010	CHECK CRD PURCHASE 10/05 ATTM*265013150797MTZ 800-331-0500 GA 425909XXXXXX3734	\$58.65
10/8/2010	CITIMORTGAGE INC LOAN PAYMT 100810 20045011523 MANGO PROPERTIES, LLC	\$1,286.76
10/13/2010	CHECK # 115	\$989.59
11/3/2010	Bus: Telephone-AT&T MOBILITY	\$114.09
11/4/2010	CITIMORTGAGE INC LOAN PAYMT 110410 20045011523 MANGO PROPERTIES, LLC	\$1,286.76
11/8/2010	ONLINE TRANSFER REF #BE29K2WZ2 TO MORTGAGE WEST VALLEY NOV 1	\$902.32
11/18/2010	CHECK # 116	\$989.59
12/2/2010	Bus: Telephone-AT&T MOBILITY	\$121.15
12/3/2010	ONLINE TRANSFER REF #BETH3QG6X TO MORTGAGE WV RENTAL DECEM	\$902.32
12/7/2010	CITIMORTGAGE INC LOAN PAYMT 120710 20045011523 MANGO PROPERTIES, LLC	\$1,286.76
12/15/2010	CHECK # 117	\$989.59
1/4/2011	Bus: Telephone-AT&T MOBILITY	\$123.15
1/5/2011	ONLINE TRANSFER REF #BEMP5GJB3 TO MORTGAGE WEST VALLEY JANU	\$902.32
1/7/2011	CITIMORTGAGE INC LOAN PAYMT 010711 20045011523 MANGO PROPERTIES, LLC	\$1,316.29
1/26/2011	CHECK # 118	\$989.59
2/3/2011	Bus: Telephone-AT&T MOBILITY	\$121.72
2/7/2011	ONLINE TRANSFER REF #BEJS8RTH3 TO MORTGAGE WEST VALLEY FEB 20	\$902.32
2/9/2011	CITIMORTGAGE INC LOAN PAYMT 020911 20045011523 MANGO PROPERTIES, LLC	\$1,316.29
2/11/2011	WITHDRAWAL MADE IN A BRANCH/STORE	\$80.00
2/16/2011	CHECK # 120	\$15.00
2/17/2011	CHECK # 119	\$989.59
3/3/2011	ONLINE TRANSFER REF #BETHKB4S4 TO MORTGAGE MARCH WV MORTG	\$902.32
3/7/2011	CITIMORTGAGE INC LOAN PAYMT 030711 20045011523 MANGO PROPERTIES, LLC	\$1,316.29
3/9/2011	AT&T Mobility 000PAYMENT 030811 00121 A 11067 # 121	\$123.72
3/18/2011	CHECK # 122	\$989.59
3/21/2011	CHECK CRD PURCHASE 03/18 RESCUE ROOTER/ARS 9151 WEST VALLEY C UT 425909XXXXXX373	\$397.95
3/29/2011	AT&T Mobility 000PAYMENT 032811 00123 A 11087 # 123	\$121.72
4/4/2011	ONLINE TRANSFER REF #BETHQNMDQ TO MORTGAGE APRIL WV MORTG	\$902.32

The next step in the procedure is to look at the data from worksheet “Raw Data” and add all the amounts from one month a year ago and throw it in the table shown in Figure 1. Then the procedure looks at the following month, adds the amounts and throws it in the next row of the table (see Figure 12).

Figure 12

Last 12 Months	Balance
May-10	\$ 2,000.44
June-10	\$ 2,175.40
July-10	\$ 2,876.59
August-10	\$ 1,654.43
September-10	\$ 5,591.60
October-10	\$ 3,259.60
November-10	\$ 3,292.76
December-10	\$ 3,299.82
January-11	\$ 3,331.35
February-11	\$ 3,424.92
March-11	\$ 3,851.59
April-11	\$ 1,975.14
Total Expenditures	\$ 36,733.64

As it is shown in Figure 12, the monthly totals are exported to the table located in the worksheet “Summary” giving the information that the user is looking for. After the amounts are exported from Raw Data to the Summary table, the amounts are shown in the graph also located in the Summary worksheet (see Figure 13).

Figure 13





## Discussion of Learning

Having never taken a VBA class prior to this semester, this project was challenging and certainly a learning experience. The goal was to create a report that would pull the entire statement from Wells Fargo and separate it by profit and expenses, including splitting expenses into categories. The challenge started from the beginning as I had trouble pulling the information from the web. The link on the web that takes you to certain reports changes every time you pull the report. So for example, if I want to see my statement and then decide to log off and then log back in to see the same report, the link of the latter is different from the link from the first one.

I could not get around to use positioning, as I had seen done in other projects, and because of my inexperience I decided to find an expenditure report that would have a fixed link, which I did. In part I was disappointed that I would not be able to have the deposits in the report to compare to the expenditures, but in part, I was glad that the report gave me part of the data I was looking for. Bringing the income into the report will be phase II of my project.

The fun part of the project was dealing with the data that the sub procedure LoadData downloads to the computer. The ProcessData sub procedure does that. The first step was to delete the extra information I was not interested in. I did that with the line of code:

```
Range("A:A,C:C,E:E").Select  
Selection.Delete Shift:=xlToLeft
```

The next part included a code to sort the data according to the date of the expenditures from oldest to newest dates. This was critical to the program in order to facilitate the adding up of the expenditures from the same date:

```
ActiveWorkbook.Worksheets("Raw Data").Sort.SortFields.Clear  
ActiveWorkbook.Worksheets("Raw Data").Sort.SortFields.Add Key:=Range("A2:A53" _  
    ), SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal  
With ActiveWorkbook.Worksheets("Raw Data").Sort  
    .SetRange Range("A1:C53")  
    .Header = xlYes  
    .MatchCase = False  
    .Orientation = xlTopToBottom  
    .SortMethod = xlPinYin  
    .Apply  
End With
```

Next I created three variables: StartDate, StartMonth, and StartYear. StartDate referred to the current date the user would be running the program minus 365, thus the StartDate becomes the exactly one year before the date the user is running the program. I did this so the program would collect date from

the oldest to the newest expenditures in the data. The next little code verified is the first date on the list was older than the StartDate, and if it was, then the entire row would be deleted:

```
Do While ActiveCell.value < StartDate
    ActiveCell.Rows("1:1").EntireRow.Select
    Selection.Delete Shift:=xlUp
Loop
```

For the next step I created two variables called RowDate and aTotalSum. RowDate is the variable that will identify what is the date for the row in which the VBA will be examining and adding the expenditure amount. The variable aTotalSum is the variable that will store the sum of amounts for all dates in the same month:

```
Do
    RowDate = ActiveCell.value
    aTotalSum(month(RowDate)) = aTotalSum(month(RowDate)) + ActiveCell.Offset(0, 2).value
    ActiveCell.Offset(1, 0).Select
Loop While ActiveCell.value <> ""
```

The last, and for me the coolest, step is where the VBA imports the data from the worksheet Raw Data into the Summary worksheet and places it into the table:

```
CurrentMonth = StartMonth + 1
Do
    If CurrentMonth < 13 Then
        ActiveCell.value = CurrentMonth & "/" & StartYear
        ActiveCell.Offset(0, 1).value = aTotalSum(CurrentMonth)
        CurrentMonth = CurrentMonth + 1
    Else
        CurrentMonth = 1
        StartYear = StartYear + 1
        ActiveCell.value = CurrentMonth & "/" & StartYear
        ActiveCell.Offset(0, 1).value = aTotalSum(CurrentMonth)
        CurrentMonth = CurrentMonth + 1
    End If
    ActiveCell.Offset(1, 0).Select
Loop While ActiveCell.value <> "Total Expenditures"
```

What the VBA code in this section does is to check if the “CurrentMonth” (CurrentMonth is a variable that identifies the month following the “year ago” month. i.e. if the report is run in June 2011, CurrentMonth becomes July 2010) is lower than 12 and then inserts the date in the table followed by the total amount for that month in the cell next to it. The code does it until it reaches 12 (December). Once it reaches 12, the CurrentMonth becomes 13 and the code jumps to Else and the new CurrentMonth becomes 1 (January). The loop continues until it reaches the row with the cell value “Total Expenditures”. Meanwhile, excel is getting data from the table and adding it to the graph previously prepared.