

## FINAL VBA PROJECT

### Description of the Business

BYU Continuing Education offers a variety of conferences and workshops ranging from youth summer programs to organ classes. Workshops and conferences are typically held on BYU campus and can be completed for either college credit or for personal enrichment.

### Issue

Any customer who has participated in a conference or workshop has a CE account with their personal information and an individualized account number. At times, two accounts will have very similar personal information; these are potentially clone accounts with that need to be merged into one account. This is a repetitive and tedious process. If either of the two accounts have a job title, then the merger cannot be completed. It is very frustrating when you click through three different pages and wait for all merge possibilities to load only to find out that it is impossible to complete the merge

### System

The system I build solves this problem by finding which potential mergers have job titles and lists these merges on an excel sheet.

**Warning: To use this system, you must have CE admin authorization.**

### The typical process

Step 1: log into the CE admin system

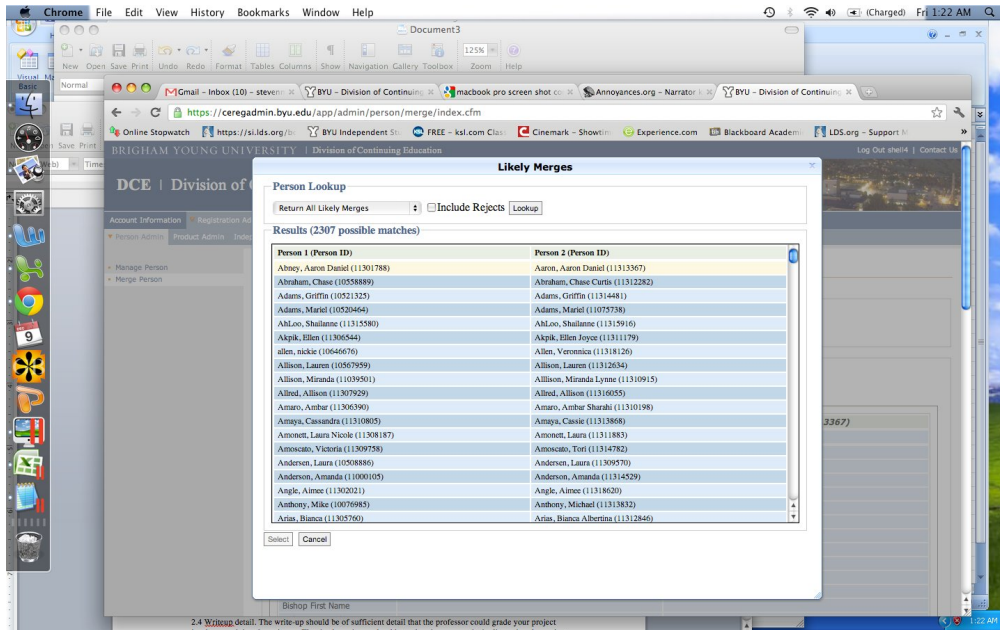
Step 2: select "merge person" option

Step 3: select "look up likely merges"

Step 4: select "look up"

Step 5: select "ok" when a dialog box asks if you want to proceed

At this point in the process we have the following list of thousands of likely merges:



We select a likely merger and the personal information of two customers with similar information are brought on screen. This is the point where you average user would be able to know if he can even complete the merger by seeing if either customer has a “job title”.

My program automates steps 1 by using an excel form.

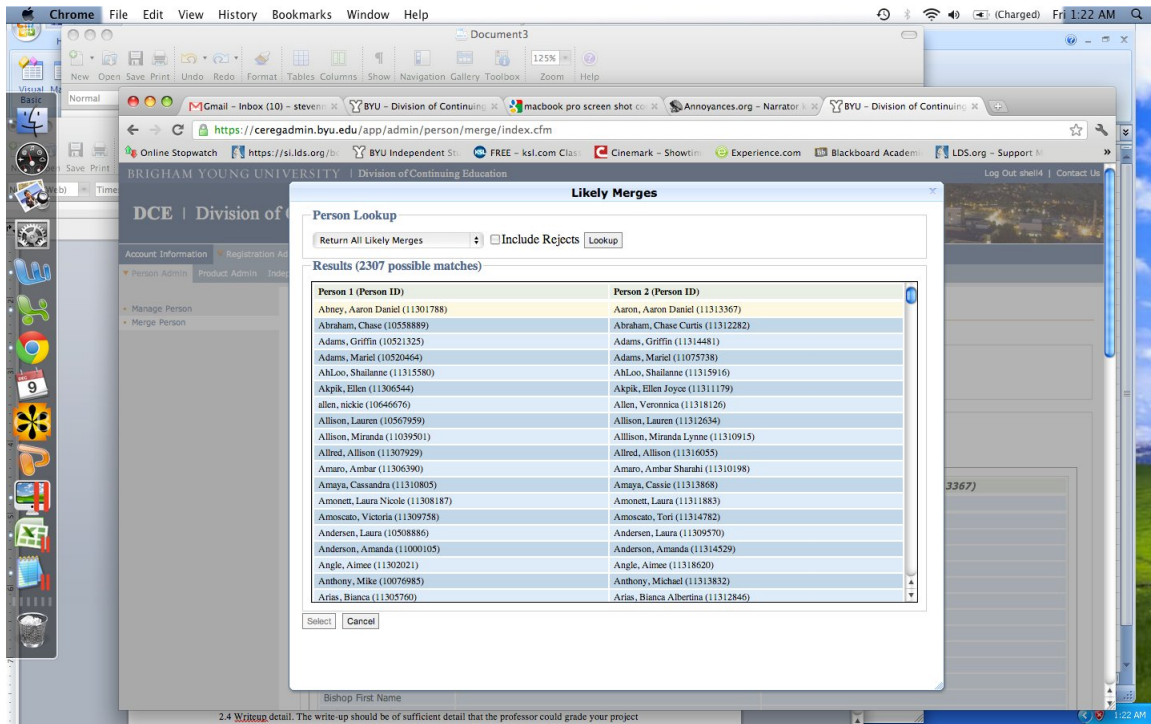
Step 2 and 3 are automated with the following code:

```
a.followLinkByText "Merge Person"
```

```
a.waitForLoad
```

```
a.explorer.document.all("LookupMergeLikelyPerson").Click
```

At this point, we have a frame open like this:



The following code disables the dialog box in step 5 and step 4 is no longer needed because the frame has been turned into the main window.

```
a.explorer.document.frames(0).document.forms(0).onSubmit = ""
a.explorer.document.frames(0).document.forms(0).submit
```

The program parses the html, which includes the two customer id numbers in the following format:

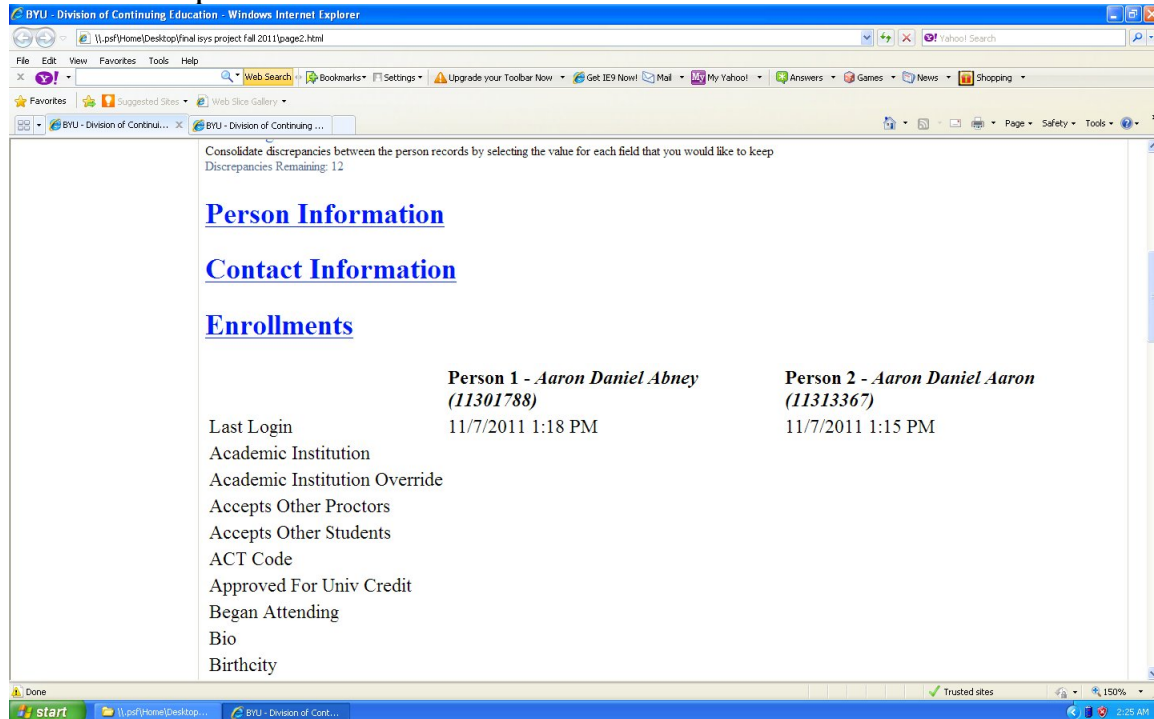
```
Pid_#####,pid_#####
```

The program uses a split function to divide these numbers and place them in a spreadsheet.

```
Data = Split(a.explorer.document.frames(0).document.all(X).ID, ",")
```

At this point the program calls the function "canMerge()"

A typical user would click on the merge option and the two customers information would come up like so:



. The url for this page is:

```
"https://ceregadmin.byu.edu/app/admin/person/merge/index.cfm?"mPerson1=pers1"&mPerson2=" &pers2&filterType=noFilter&letterMatch=&IncludeRejects=false
```

With pers1 and pers2 being = to the first customer number and the second customer number respectively.

These values of With pers1 and pers2 are pulled from the excel sheet that we created in sub merge() -which is a clever name- procedure and the needed url is created.

The program saves the url, and the source is parsed first for a job title from the first customers information and puts the value in column c of the spreadsheet next to the respective customer numbers.

Then, it searches for a job title from the second customer information and puts that value in column d of the spreadsheet Next to the respective customer numbers.

This was done with the following code:

```
Do While a.moveTo("JOB_TITLE")  
a.moveTo "JOB_TITLE_1">
```

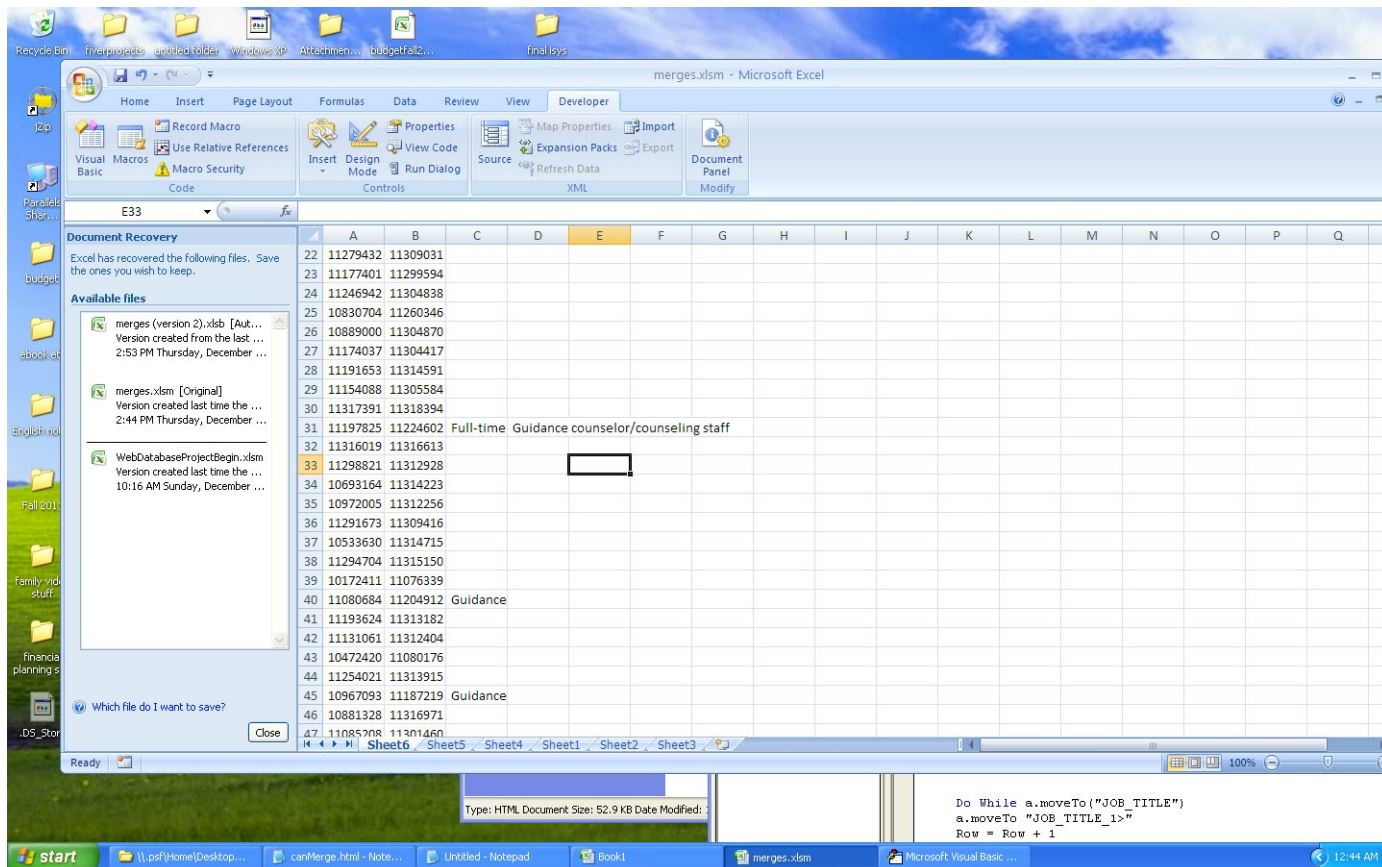
```
Row = Row + 1
jobTitle1 = a.getText("</TD")
```

```
a.moveTo "JOB_TITLE_2">"
Row = Row + 1
jobTitle2 = a.getText("</TD")
```

```
t = t + 1
Cells(t - 1, 3).Value = jobTitle1
Cells(t - 1, 4).Value = jobTitle2
```

...and now we start over.

The loop gives us new values of pers1 and pers2 and creates another url to parse until the all potential merges have been parsed. I set the loop to just 45 even though there are typically a couple thousand potential merges. The final result looks like this:



The final result is a list of the customer numbers in column A and B and their respective job titles in column C and D.

Now employees can know exactly which merge possibilities cannot be completed in a matter of minutes.

### What did I learn?

I learned a lot about html. I ran into some difficulty submitting the username and password because byu changed their website recently so the code I had wouldn't work. Instead of clicking a button like before, I had to submit the form.

I learned about frames in html and how to make a frame the primary window. Before this, I didn't even know frames existed.

I have a much better understanding of parsing html. The two tags were different (JOB\_TITLE\_1>" and JOB\_TITLE\_2>" so I had to get creative. At first, the program listed them in just one column and they weren't lining up with their actual merges. But I fixed this.

I learned better how to follow links and "click" won't always work. I learned how to use a split function which was very cool.