Kaden King

April 14, 2015

IS 520

Final Project

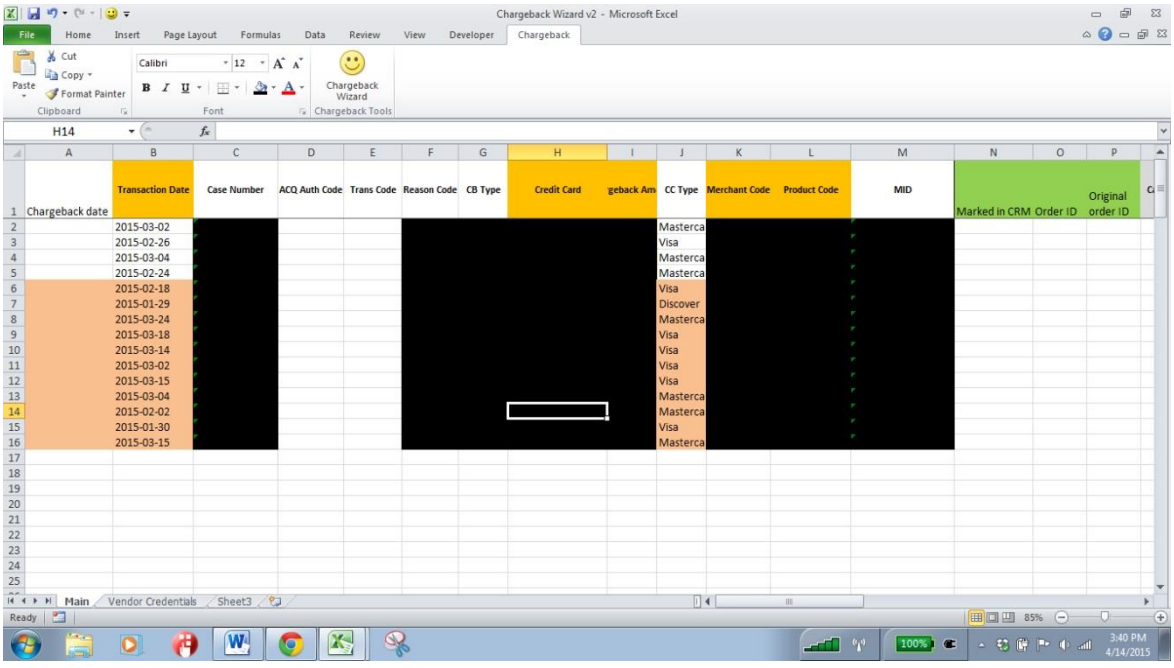
Executive Summary

My sister works as a data-entry employee for Bankable Payments, a credit card chargeback company. When there is a dispute over a transaction or a customer wants their money back, this company checks each dispute to see whether or not the transaction has broken any rules set by VISA, MasterCard, Amex, or Discover.

There are several steps in this process. The tasks that have been assigned to my sister often involve a lot of repetitive data entry. This process is prone to inaccuracy, and is time-consuming. My final project was to write a program that would automate as much of her job as possible.

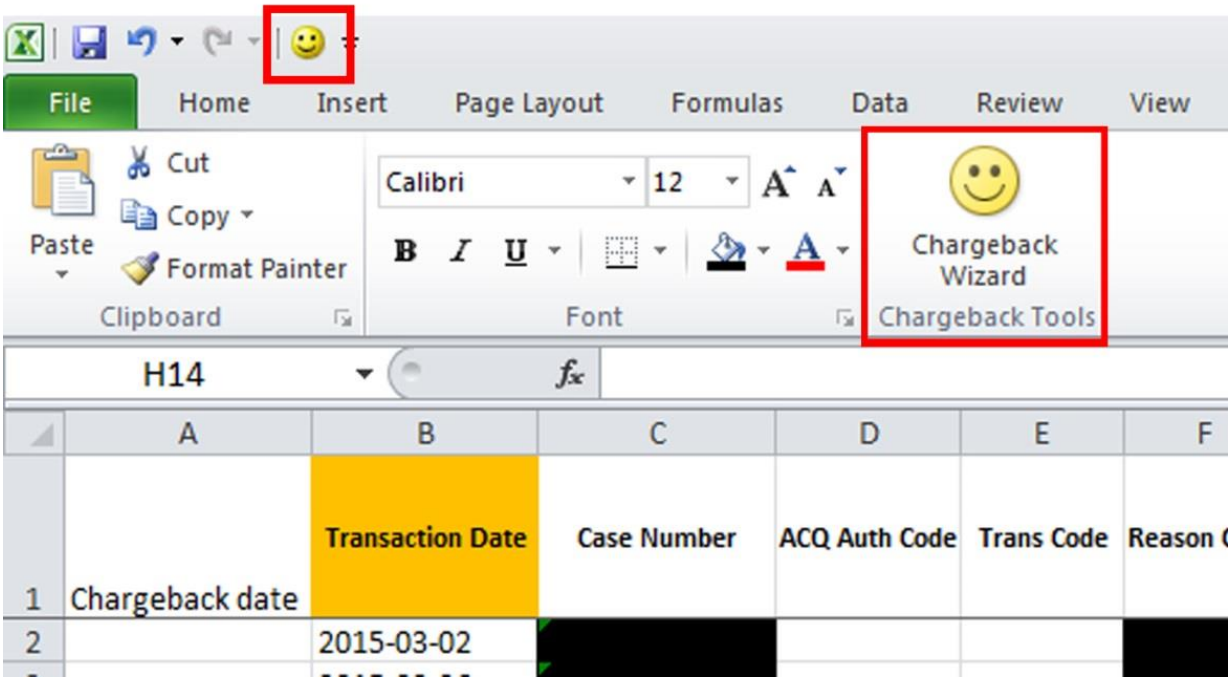
Implementation Documentation

My sister works in several spreadsheets. My program is confined to a separate workbook in order to avoid making changes (particularly unwanted or unexpected ones) in the master shared sheets. She copies records in, runs the program, and copies the results to the master copies. ****fields have been censored for privacy and legal reasons****



The first step in the program is a graphical user interface. This form allows the user to make choices about the program, and makes it simpler to run it.

The interface is accessed through a custom-made button on the ribbon. In my sister's case, I was unable to edit the ribbon, and wasn't able to find out the problem in time. Therefore, I made it available on the quick-access toolbar for her. The chosen icon reflects her feelings about the program.



Interface:

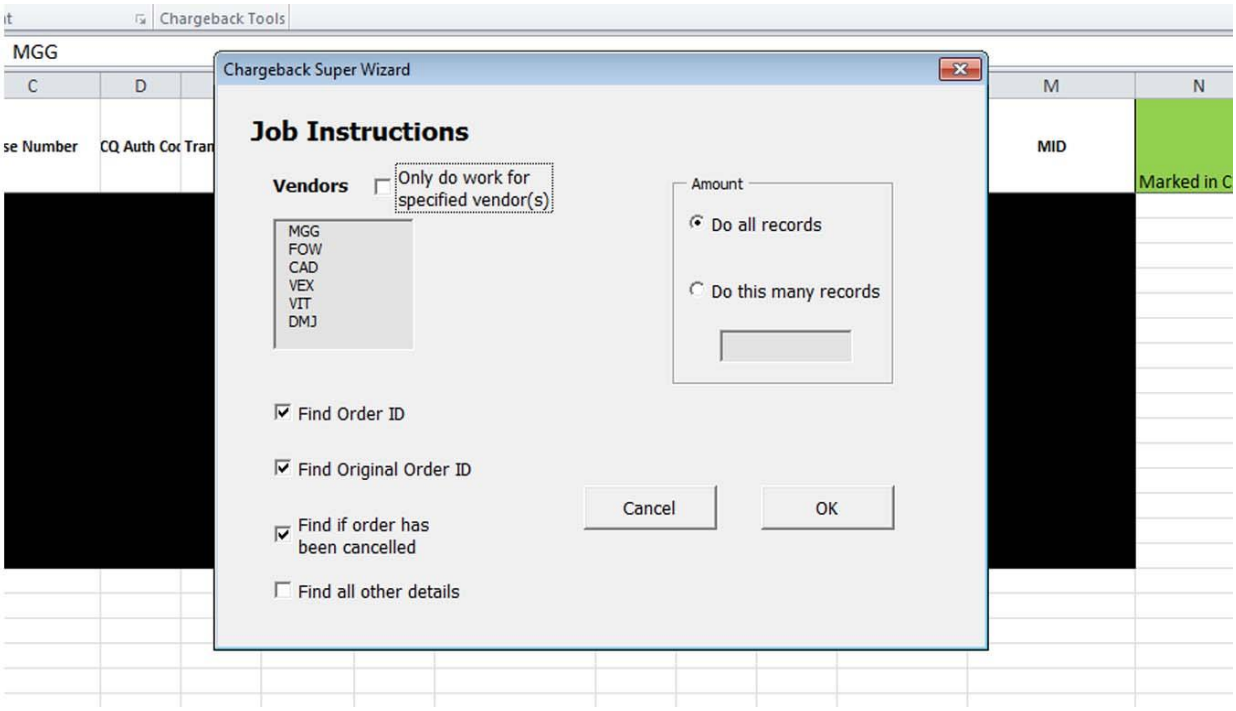
-Allows the user to choose certain vendors from a list, or to complete records for all vendors

-Allows the user to choose which information they would like the program to retrieve

-Allows the user to specify a number of records to do

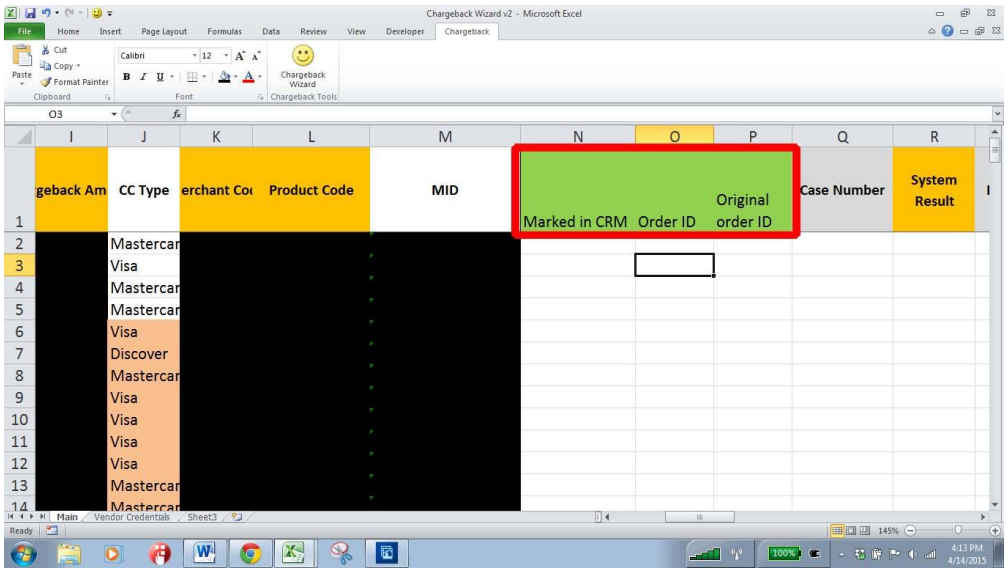
The cancel button will close the form

The "OK" button collects the user's choices, and begins the next part of the program

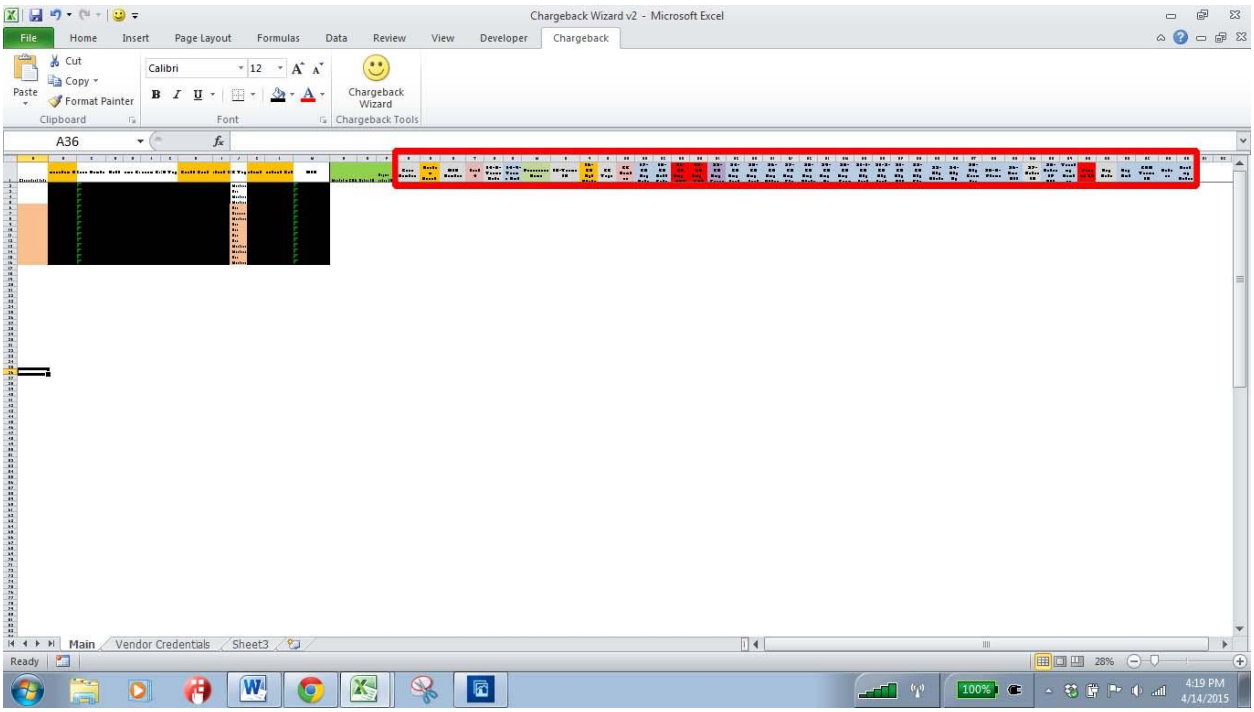


When the "OK" button is clicked, the program stores the checkbox values. If the user specified a certain vendor(s), the program loads them into an array so that the program will stop when it's done all the records. If the user didn't specify, the program will detect which vendors have records to be processed. If an amount of records was specified, the program stores it. Otherwise, it assumes that the user wants all the records done.

The first three tasks listed ("Find Order ID", "Find Original Order ID", "Find if order has been cancelled") were initially the tasks that my sister was responsible for. Each box refers to one field/column on the spreadsheet.

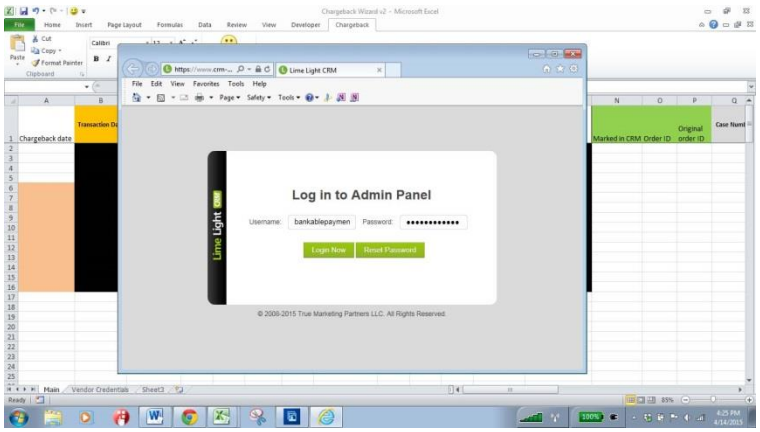


However, after my sister started using the program she was given a significantly larger amount of work. After all, she was completed her work faster, so the fourth box was added to include all the other transaction details.

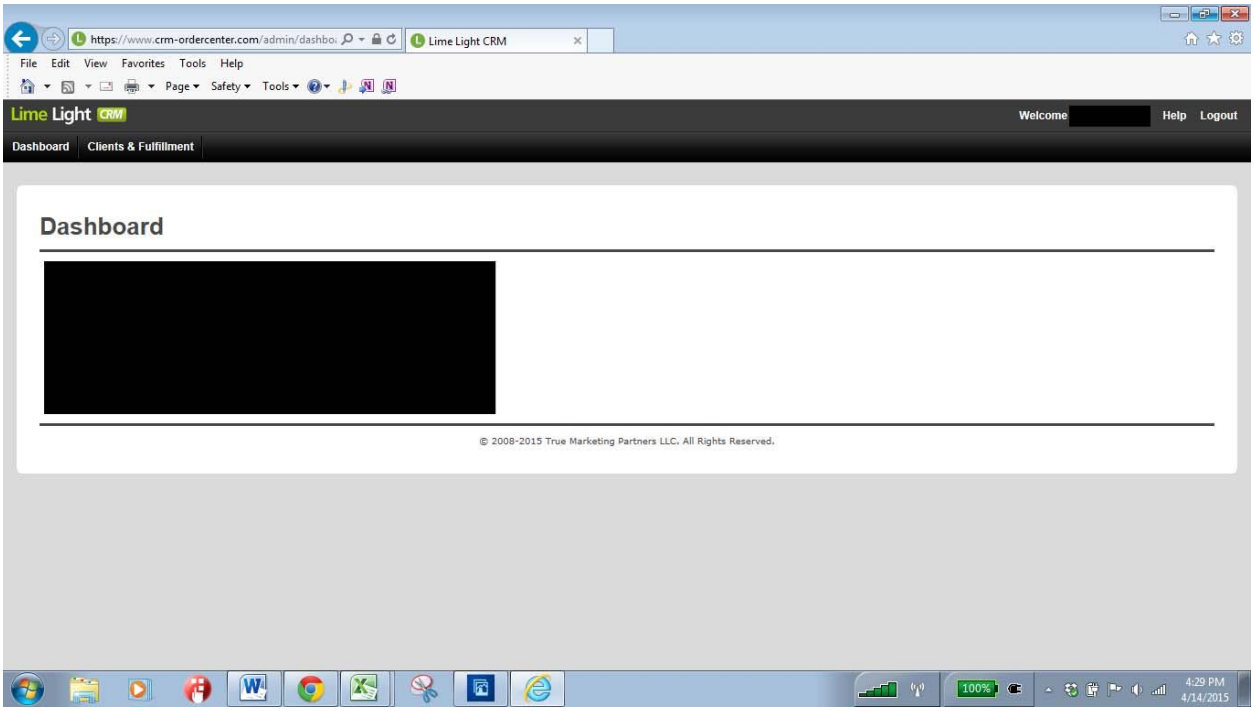


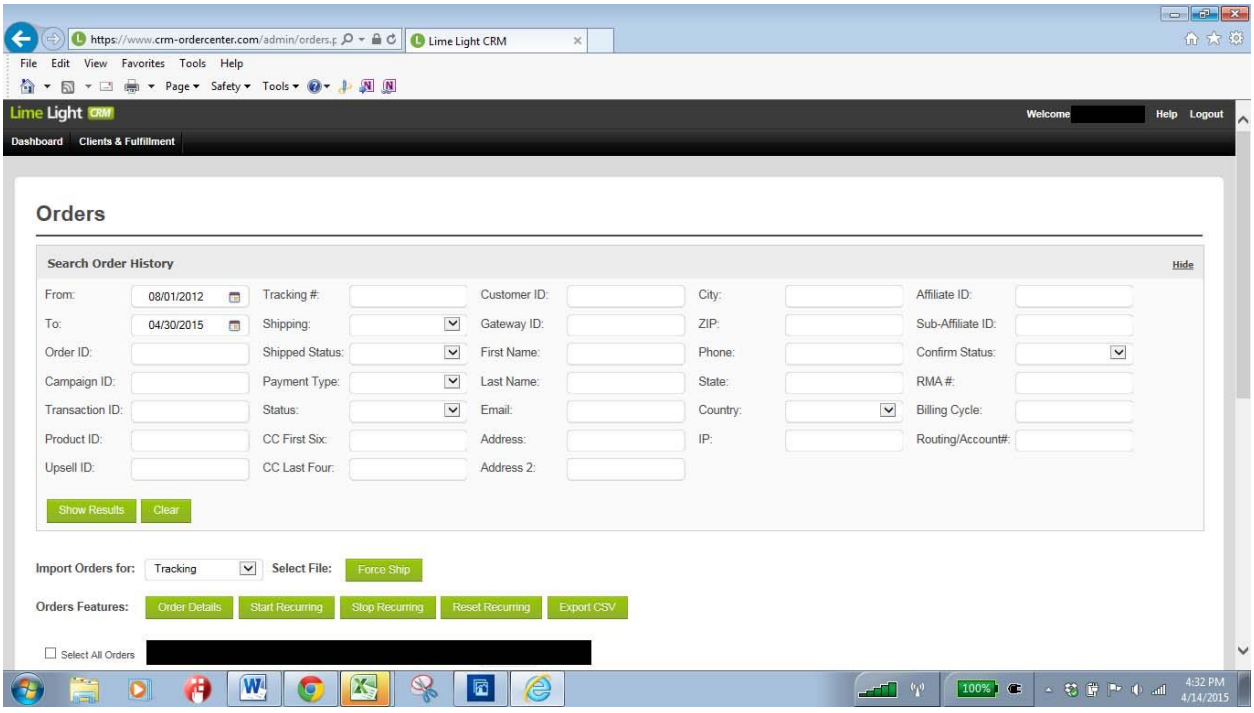
Each vendor has their own URL, username, and password that it uses to access its records. Those credentials are kept on a new sheet called “Vendor Credentials”. Instead of keeping them on random documents, all the credentials go here. The procedure starts with one vendor, and then does all the records for that vendor before moving on. This prevents the program from having to login and logout for every record, wasting valuable time.

The login page looks the same for each vendor, but the URL, username, and password is different. The credentials are used to login to the first vendor. A basic error trap is used in case the user is already logged in, in which case the “Login Now” button is absent, which causes an error to occur.



This takes you to the dashboard. From here, it will navigate to the “orders” page.

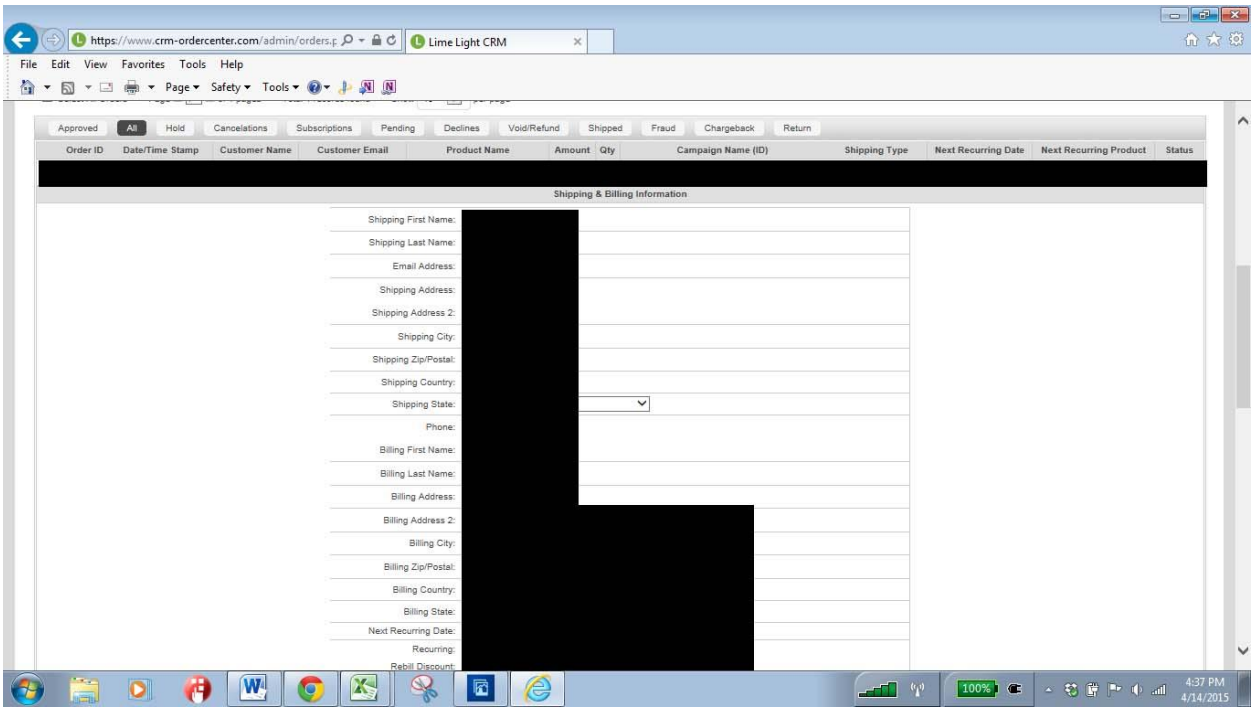




The first 6 and last 4 digits of the given credit card number are used to find a customer’s records. Then, the transaction date and amount are used to find the specific purchase.

To access these values, the order page is imported as a new page, where the correct values are found. Once the correct order is found, the link associated with that order is opened.

This leads to a page with all the details for that order.



Many of the fields found here are copied to their respective fields on the spreadsheet.

If there is if there is no tracking number, the program needs to find the earliest record with the same product code as the first one it found. Sometimes it's hard to find the right order because the amount is split between a base amount and a shipping amount. If the procedure is unable to find the order it's looking for, it parses out the shipping amount and adds it to the base, and looks through the orders again. If it's still unsuccessful, it will leave the record blank and iterate to the next row. These are unusual circumstances that require human interpretation. They are relatively rare, and the program usually will skip fewer than 1 in 20 or 1 in 30 of the records.

If the tracking number is not present, then all of the values for the "Original Order" as it is called in the spreadsheet, are copied from the "Order". If it is present, the program finds the Original Order (as described above). This page is also imported, and several more fields of data extracted.

There are some error traps that anticipate problems such as a missing field/label.
When one vendor is completed, the program logs out the user, and proceeds to the next website. At the end of the program, both imported pages are deleted, and the user form is closed.

Learning and Difficulties

It took me a long time to really start to understand how to use the agent class to interact with the HTML. I eventually figured out how to find a tag and use it. However, I was completely defeated when I had to figure out how to use a link. I was caught up in "tag-finding" and forgot the obvious fact that a link is simply a URL, and can be "clicked" with the openpage method. I had to go in for assistance from Professor Gove Allen, because there aren't any online help forums for the agent class. He explained how to follow the link. While I was there, he also helped me learn how to compare dates, and use logic to find the value I was seeking.

I learned a ton about agent. I learned a lot about debugging – my sister found several problems where the program inserted the wrong values, or simply stopped running because of an error. I know that I haven't written enough error handling into the code. I've realized that a lot of things can go wrong, and you have to anticipate some errors for every step in the code process. This means that I have to protect many fields from errors caused by a missing value, or a type mismatch. I need to write a something that will identify if a column header in the home spreadsheet is missing or misspelled. Ideally, something that will warn the user and ask them to either specify a column, or skip it.

Aside from Professor Allen, I got the rest of my help online.

Conclusion: With the first three fields I automated, I have been able to cut 1-3 hours of work out of every single day for my sister. With the new fields that have now been assigned to her, my sister was able to complete 12 records in an hour. This program can do that many in less than 5 minutes. Not only have I freed up her schedule significantly (decreasing stress as well), but she pays me her wage for 1 hour out of every day.

This program has the potential to greatly decrease the time it takes to complete this particular task. Given the time saving, there is potential for me to sell this program to the owner of the business for a substantial amount.