

# EFTPS TIME SAVER

## EXECUTIVE SUMMARY

Like many other students and employees, I do all I can to avoid time-consuming, mind-numbing activities. One such type of activity is paying payroll taxes for the small business for which I perform payroll duties. A typical Thursday afternoon I could be found clicking through the Electronic Federal Tax Payment System website for up to 15 minutes entering information piece by piece as I navigated through the payment submission process. Often, I finished only needing to return to the beginning to repeat the process again to process additional payments. This project attempts to automate this work.

The program I have written first accepts login credentials from the user, then logs into the EFTPS website. It then retrieves options from the webpage and asks the user to input all of the payment information in a second user form. This allows all of the payment information to be entered using only one form and saved for later entry into the website. When the form is saved, a number of data checks are run on the inputs to ensure that the program can run without throwing an error from the website. The program then navigates through the system, entering the relevant information where necessary. When all of the information is entered, the program displays the confirmation page from the webpage and asks the user to verify the information. If it is correct, the payment is submitted and the user has the option to make another payment or quit. If the information is incorrect, the user can reenter the information. After the payment is submitted, the information is saved into the Excel workbook along with the confirmation number from the EFTPS website and the Internet Explorer is closed. This program typically now only takes about 1 minute to run, depending on the speed of the internet connection.

## PROJECT EXPLANATION

### FORMS

My original goal in creating this program was to provide a solution that would allow me to enter all of the necessary information in a centralized location to avoid having to click through the Electronic Federal Tax Payment System (EFTPS) and enter information piece by piece. So, I started by creating the forms for data entry.

The first form I created was the payment form (Figure 1). I created text boxes for the payment amounts, settlement date, tax payment type,

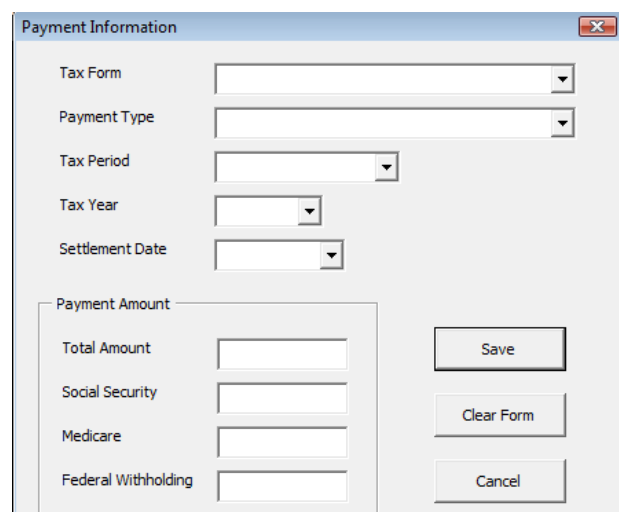
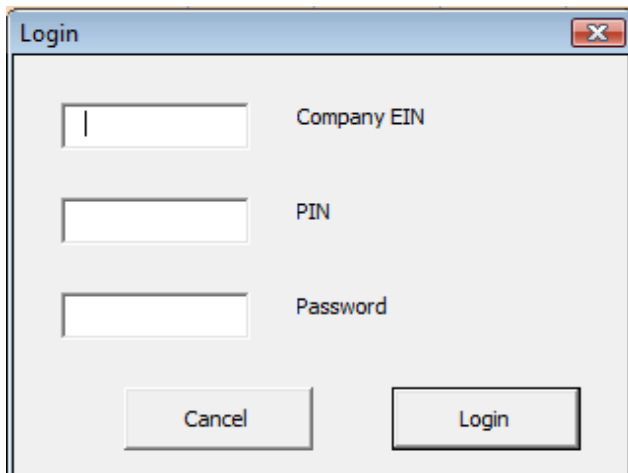
The image shows a screenshot of a software window titled "Payment Information". Inside the window, there are several input fields: "Tax Form" (a dropdown menu), "Payment Type" (a dropdown menu), "Tax Period" (a dropdown menu), "Tax Year" (a dropdown menu), and "Settlement Date" (a date picker). Below these fields is a section titled "Payment Amount" which contains four text boxes labeled "Total Amount", "Social Security", "Medicare", and "Federal Withholding". To the right of these text boxes are three buttons: "Save", "Clear Form", and "Cancel".

Figure 1 - Payment Form

etc., and then realized that I would need some way to ensure that the information entered was correct. I then decided to use combo boxes for the information that had only a set number of possibilities, to ensure that the information entered was correct. One difficult combo box was the tax form box. I knew the current types of tax forms available, but I wanted to make the program such that it would be more robust if the website were to change. Thankfully, the website had a similar combo box that I was able to extract information from by learning the properties of combo boxes in HTML. I used the websites combo box to populate the box in the form. The settlement date was also a little tricky because the website would throw an error if a non-business day was entered, so I had to learn the functions to ensure that a business day was selected (I haven't figured out how to deal with holidays, yet).

Once I had all of the boxes created, I added functionality to the buttons on the form. The cancel button unloads the form and then asks if the user wants to leave the program. If so, the sub ends and a public variable is set to true. When the main calling sub takes back over, it runs an if statement and ends the whole program if the quit variable is true. The clear form button merely clears all information entered into the form. The save button takes all of the data and runs a few data checks. First, it checks all combo boxes to ensure that an option has been selected and then saves the entries to public variables for use in the module. Next, it checks all the text boxes for entries, ensures that the entries are numeric, compares the entries to the proper format, and then saves the entry as a public string variable to be entered into the website later. It also sums the sub category entries to make sure they equal the total entered. Finally, the form is hidden and the user sits back while the rest of the code executes.



**Figure 2 - Login Form**

The second form I created was the login form (Figure 2). After watching Professor Allen walk through logging into the route Y website, this part was pretty simple. I created text boxes for the EIN, PIN and Password, and then set the password character for both the PIN and Password textboxes to an asterisk for privacy. Next, I created the buttons to allow a user to cancel (this button uses similar logic to the cancel button on the payment form) which asks him or her to confirm and then exits the whole program, if desired. The login button

runs a few data checks to make sure the boxes contain an entry, and that the entry is numeric and the proper length. The button then saves the entries into public variables to allow the main sub to log in to the EFTPS website.

## SUBS

In order to keep the procedures manageable and callable from other subs, I created a fairly involved set of calls to subs. An outline is as follows (lower levels indicate a call to a sub from the higher level sub):

- runAll
  - login
    - login form
  - enterPayment
    - moveToPayment
    - payment form
    - enterForm
    - selectPaymentType
    - enterTotal
    - enterCategoryAmounts
  - verifyPayment
    - enterPayment
    - savePaymentData
  - addPayment
  - closeBrowser

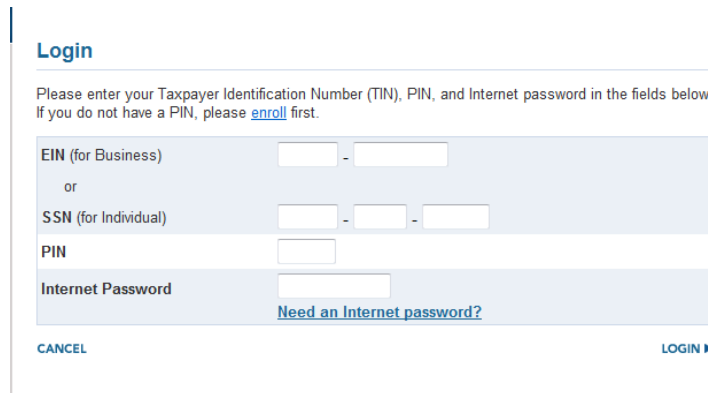


Figure 3 - EFTPS Login Form

The runAll sub is basically the master sub that calls all of the other subs. It does have some functionality in parts where the user can request to cancel the program and exit. This sub also manages the protecting and unprotecting of the worksheet (nothing sophisticated, just unprotects it and protects it when exiting or ending the sub). I decided to add the protection to the worksheet to prevent accidental deletion of data.

The login sub takes care of opening the login form (described above), navigating to the login page (Figure 3), and entering the login information. The values are taken from the login form and entered into the appropriate fields. This sub also attempts to deal with the problem of incorrect login credentials.

The enterPayment sub calls the payment form and the subs necessary to enter the payment information in the successive pages on the EFTPS website. It also handles the situation of a user requesting to leave the program after clicking cancel on the payment form.

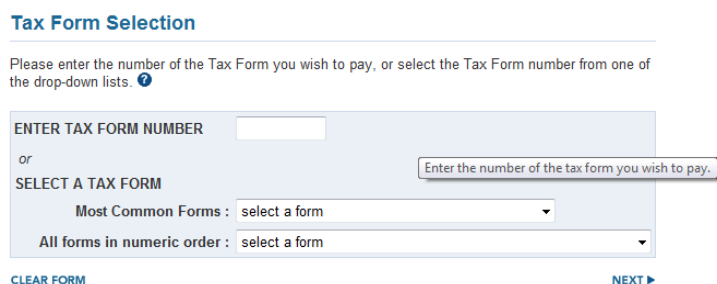


Figure 4 - EFTPS Tax Form Selection Page

The moveToPayment sub moves the browser to the tax form selection page. The enterForm sub then makes the appropriate selection in the “Most Common Forms” combo box (This is also where the payment

user form pulls the information to populate the “Tax Form” combo box). The sub then clicks the next image to advance.

The selectPaymentType sub takes the tax payment type selected in the payment form and selects the corresponding tax payment type on the EFTPS tax type selection page.

This sub also checks to be sure that the radio button description matches the selection by the user (this helps catch any changes in the EFTPS ordering of the tax payment types). The page is then advanced to the next step.

### Business Tax Payment

If you select the next business date for your settlement date, you will not be able to cancel this payment. EFTPS requires your settlement date to be at least 2 business days in the future to cancel a payment.

Please enter the payment amount in the following format: \$\$\$\$\$\$.##

For fiscal year taxpayers, please enter the four-digit year in which your tax filing period ends. [Click here for an example.](#)

Payment Amount	\$	<input type="text"/>	(example: 1234.56)	?
Tax Period	Month	December (4th Quarter)		?
	Year	2010	(yyyy)	
Settlement Date	12	/	10	/ 2010 (mm/dd/yyyy)

Enter the Settlement Date for the payment

CLEAR FORM ◀ PREVIOUS | NEXT ▶

Figure 6 - EFTPS Business Tax Payment Page

string, every figure is entered. Similarly, the date and year needed to be entered with the appropriate number of digits for the website to recognize the entry as valid. Thus, the data checks in the payment form help to ensure that the website doesn’t throw an error when this sub clicks the next button to advance to the next page.

The enterCategoryAmounts sub again takes the information entered in the payment form and enters the category amounts in the appropriate boxes in the EFTPS Sub Category Amounts page. This page also necessitated the data checks in the payment form. The amounts entered in the sub category fields must be in the same number format as the total on the previous page, and must also add up to the total amount. By checking the data before entering it into the webpage, I was able to prevent the page from throwing an error and thus causing the program to throw an error.

### Tax Type Selection

Please select a Tax Type: ?

☐ Federal Tax Deposit

☐ Balance due on return or notice

☐ Payment Due On An Amended Or Adjusted Return

☐ Audit Adjustment

◀ PREVIOUS | NEXT ▶

Figure 5 - EFTPS Tax Type Selection Page

The enterTotal sub enters the information from the payment form into the appropriate boxes on the EFTPS Business Tax Payment page. This page is the reason I needed to perform data checks on the information entered into the payment form. The total amount needed to be in the following number format: \$\$\$\$.\$\$. This required the amount to be entered as a string rather than as a value because any zeros on the end of the figure to the right of the decimal are dropped in numeric format, while as a

### Sub Category Amounts

For the tax form you have selected, please break down the amount being paid into one or more of the following Sub Categories. The total of Sub Category amounts must equal your Payment Amount. ?

PLEASE NOTE	
Any amounts represented in the subcategories of Social Security, Medicare, and Income Tax Withholding are for informational purposes only.	
Tax Form Selected	
Tax Form	941 Employers Federal Tax
Tax Type	Federal Tax Deposit
Payment Amount	\$1,000.00
Sub Category Amounts	
1 Social Security	\$ <input type="text"/>
2 Medicare	\$ <input type="text"/>
3 Tax Withholding	\$ <input type="text"/>

CLEAR FORM ◀ PREVIOUS | NEXT ▶

Figure 7 - EFTPS Sub Category Amounts Page

## Verify Payment Information

Please review all the information you have input before you click "Make a Payment." If you wish to make changes, click the "Previous" button below.

**PLEASE NOTE**

Any amounts represented in the subcategories of Social Security, Medicare, and Income Tax Withholding are for informational purposes only.

Payment Information	Entered Data
Taxpayer EIN	xxxxx4195
Tax Form	941 Employers Federal Tax
Tax Type	Federal Tax Deposit
Tax Period	December/2010
Payment Amount	\$1,000.00
Settlement Date	12/20/2010
Subcategories:	
1 Social Security	\$500.00
2 Medicare	\$250.00
3 Tax Withholding	\$250.00

[← PREVIOUS](#) | [MAKE PAYMENT](#)

Figure 8 - EFTPS Verify Payment Information Page

The savePaymentData sub writes the payment information to the excel spreadsheet (Figure 9). It also attempts to resize the column, but this part has not worked consistently for me.

The addPaymentData sub allows the user to enter additional payments. It basically runs the

	A	C	D	E	F	G	H
1	Payment Type	941 Employers	941 Employers	941 Employers	941 Employers	Federal Tax	
2	Settlement Date	11/29/10	12/08/10	12/03/10	12/14/10		
3	Total	4,144.30	1,000.00	3,610.80	1,000.00		
4	Social Security	2,802.80	500.00	2,176.70	500.00		
5	Medicare	655.50	250.00	509.10	250.00		
6	Withholding	686.00	250.00	925.00	250.00		
7	Confirmation Number	00045680	00399823	00137830	00111624		
8	Cancellation Conf.		13274960		04150629		
9							
10							
11							
12							
13							
14							
15							
16							
17							

Figure 9 - Excel Worksheet With Saved Data

The verifyPayment sub requests the user to verify the payment information before allowing the program to submit the payment. If the payment information is false, the full enterPayment sub is run again to allow the user to edit the information. The user also has the option to cancel and confirm the request to leave the program. This allows for one last chance to end the program before submitting the payment to the EFTPS website. Finally, this sub saves the data using the savePaymentData sub, submits the payment, and saves the confirmation from the following page.

program again until the user says that no additional payments are necessary.

Last of all, the closeBrowser sub clicks on the logout button and handles the additional pages needed to successfully log out of the EFTPS website.

## CONCEPTUAL DIFFICULTIES AND LEARNING

As with all projects, I encountered a healthy dose of difficulties in creating this program. One big problem I had was that the "Next" button on the EFTPS website (and in many of the figures in this document), did not have a way to reference it, nor did the forms that I was trying to submit. I struggled with this for a time until I asked Professor Allen if he knew of a way to get around this. We worked together and eventually came to the solution of writing an additional procedure in the class module that would allow me to click on an image by supplying the url of that image. This sub cycles through all tags in the webpage and compares the url supplied with that of the current tag.

When it finds a match, the sub performs the click method of that tag. The sub has worked beautifully for me.

Another difficulty for me was figuring out how to get the total and sub category amounts in the proper number format. Though it was a relatively simple concept, I learned the important difference between a string variable and a numeric variable (integer, long, single, double). I also learned that I could use both the value property and the text property of a text box to check if the sub categories sum up to the total, then save the text as a string with all of the necessary zeros.

Finally, I had a difficult time figuring out how to get a handle on the confirmation number in the confirmation page of the EFTPS website. There was no identifier that allowed me to pull it down and save it. I asked some classmates and searched on the internet for a way to identify it, but had not luck. Finally, I decided to browse through the class module that Professor Allen provided us (it provides much of the functionality in working with Internet Explorer). There, I found a procedure that allowed me to look at the html source as text and move around in it. From there, I realized that the confirmation number was the only bolded text on the page. I was then able to move to the html tag <B> and bring back the appropriate number of characters to retrieve the confirmation number.

In sum, not only did I learn more about VBA and Excel, I also learned how to work with html and objects in Internet Explorer.

## CONCLUSION

I had a good time working on this project. Though it will take a good number of Federal Employer Tax payments using this program to justify the time it took to write it, I learned a great deal that will be helpful to me in the future; specifically, the ability to see an opportunity to improve on a process and the skills to work through the solution systematically using available resources. I hope that this program will benefit not only me, but others who may find a use for it as well.