ELEC 5220 Information Networks and Technology  
Lab 3 Report

Robert Skelton

# Abstract

The main point of this lab was learning Javascript to implement functions that could verify input on a webpage using Regular Expressions. We also built upon our foundational PHP and HTML knowledge, and were introduced to XML, or Extensible Markup Language.

# Introduction

For this lab, I used the same setup as in previous labs. I am running the Apache server on my Windows 7 Ultimate 64-bit OS, using Google Chrome as my internet browser and Sublime Text 3 to modify code. I had seen a little Javascript before this lab from looking at Codecademy, but this Javascript we wrote this week was much more intense.

# Design

**Validating User Input**

For this step, we were to create a website that could validate user input before pressing submit on a form. For this, we were introduced to Javascript, which can run on a page without user input, and Regular Expressions, which are complicated but useful ways to verify input. For all the Regular Expressions (regex), I used a Regex Generator online or Stack Overflow, because it is pretty much impossible to memorize how to write Regular Expressions. Anyways, the page we were asked to write would take user inputs of First and Last Name, Phone Number, Full Address, Zip Code, and Birthday. The Javascript would verify that each input was in the correct format, and if not would issue a popup telling you to fix your input. When you hit submit, the information is printed out on screen and then stored in a text file called userprofile.txt in XML format. XML is a markup format, similar to HTML. You can see the step by step below on what happens when you input text to the website, called 5-f.html. 5-f.php echoes all the information inputted, and then stores it to the XML formatted text file. Dr. Wu provided lots of good code to start with, and implementing the rest of it was not too difficult. We used the onblur event in JavaScript to verify all input.

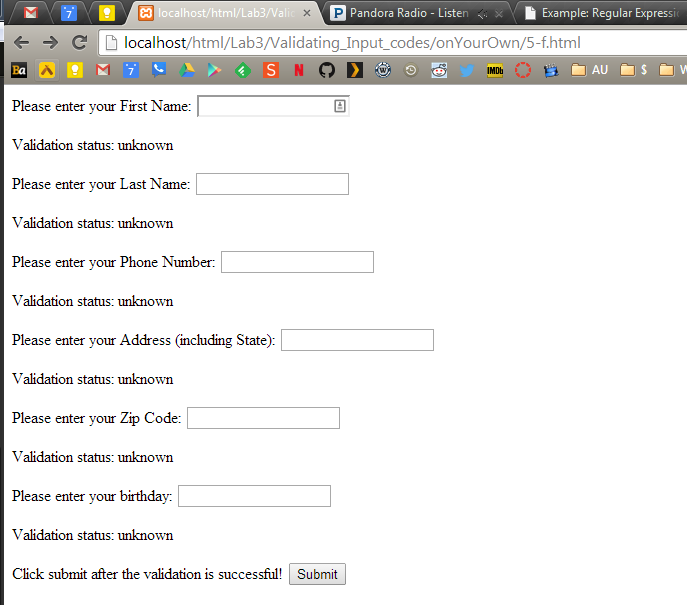


Figure 1. Before any input is submitted through the form.

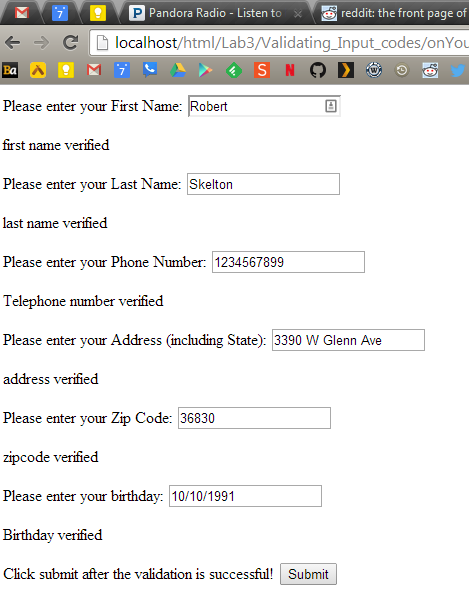


Figure 2. All input is verified using Javascript and regular expressions.

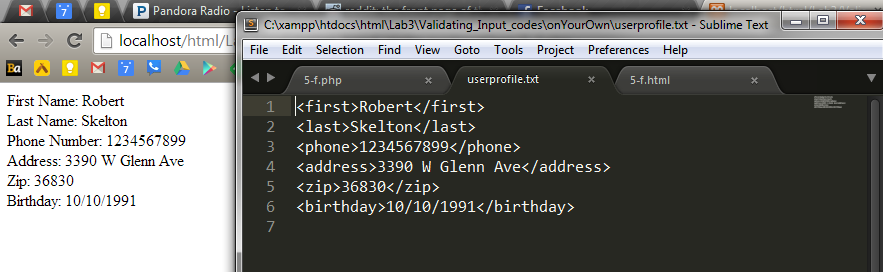


Figure 3. Verification Screen after submitting information and userprofile.txt file that stores the information in an XML format.

**Credit Card Check**

For this step, we were to combine two given php and HTML files that would check for a Credit Card’s expiration date and then verify that the Credit Card number is valid using the Luhn algorithm. Luhn is an algorithm that all credit card brands follow, and each card brand is different. We had to implement the Luhn algorithm ourselves for American Express, which meant the first digit had to be a 3 and the second digit had to be a 4 or 7, and there had to be 15 digits in the card. This was simple enough to implement by modifying the code we were given for this lab. I ran each set of valid credit card numbers through my webpage and they worked correctly, and the American Empress algorithm I implemented worked as expected. I then had to redo everything using Javascript. I used lots of the base code from the Verifying Input Code to write the Javascript for this section.

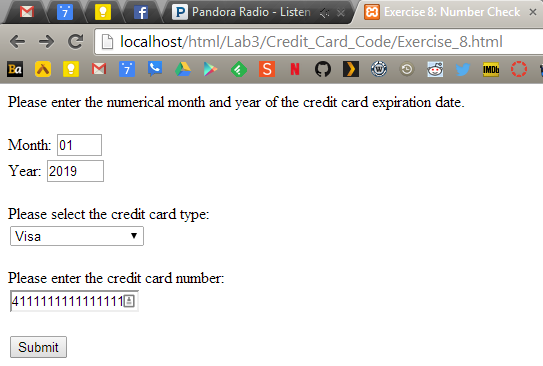


Figure 4. Using a test Visa Card that is not expired.

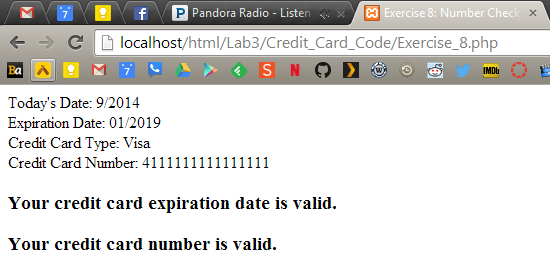


Figure 5. The Visa Verification returned a good, non-expired card.

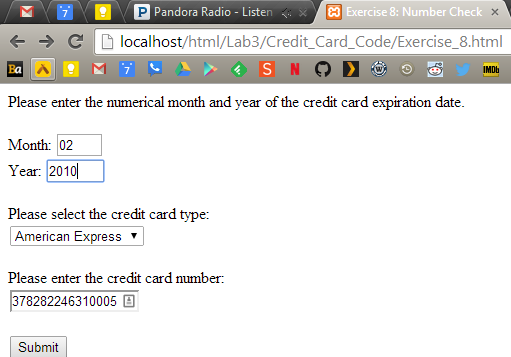


Figure 6. American Express Test using an expired card.

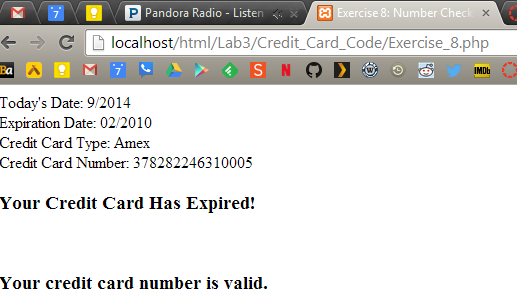


Figure 7. American Express Verification returned a valid but expired card.

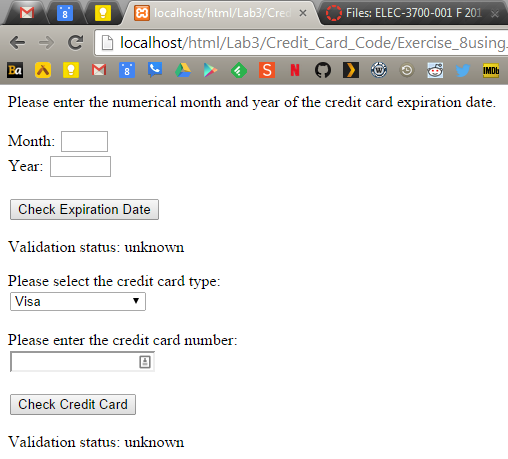


Figure 8. Before inputting any information to the Javascript form.

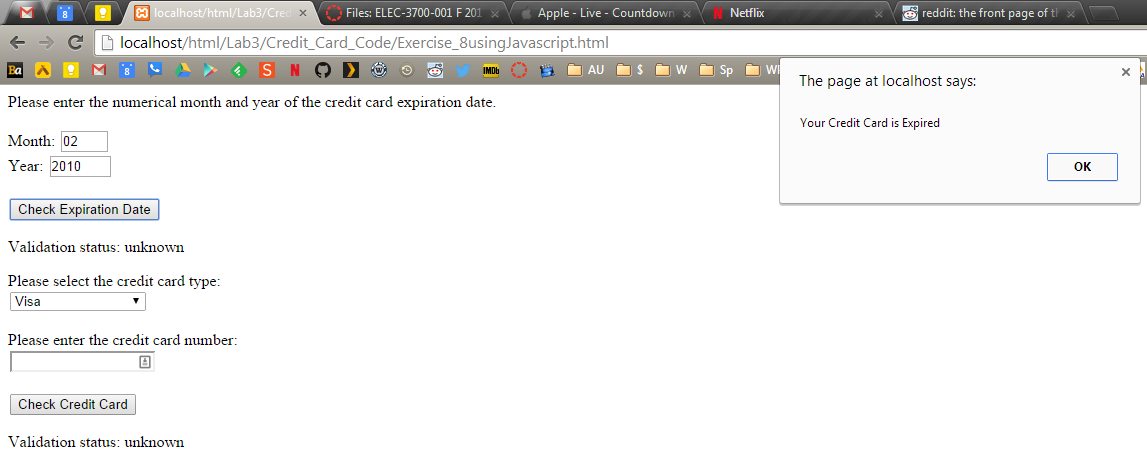


Figure 9. Checking an expired Credit Card.

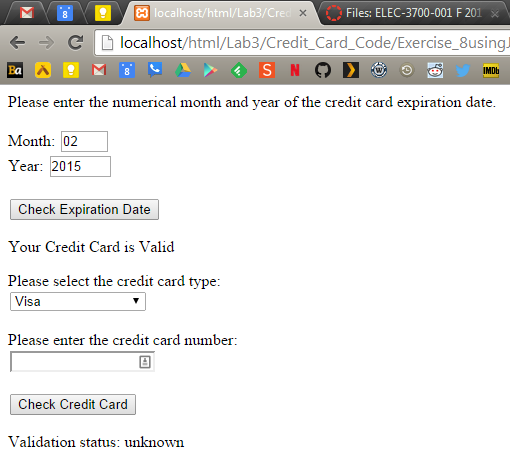


Figure 10. Verifying a valid Credit Card expiration date.

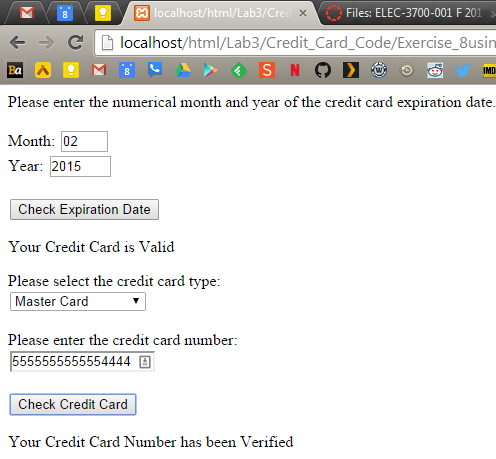


Figure 11. Verifying Master Card.

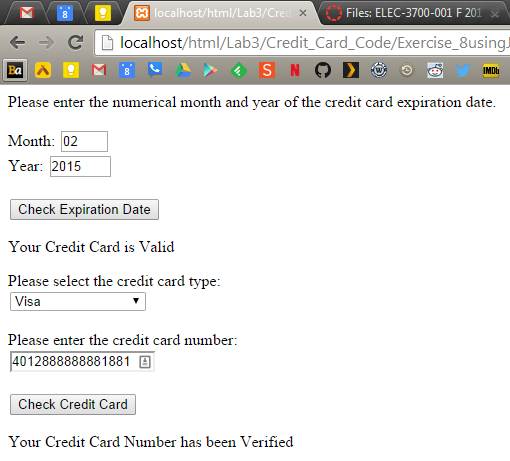


Figure 12. Verifying Visa.

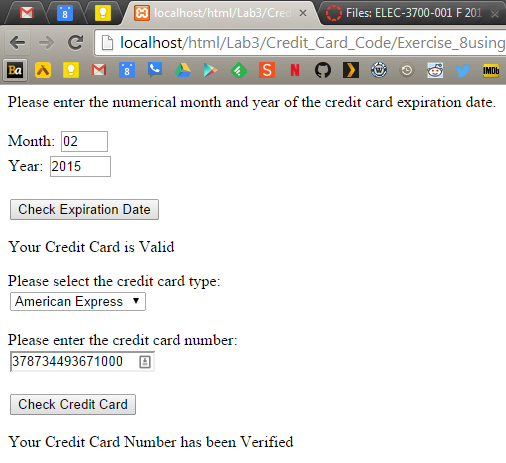


Figure 13. Verifying American Express.

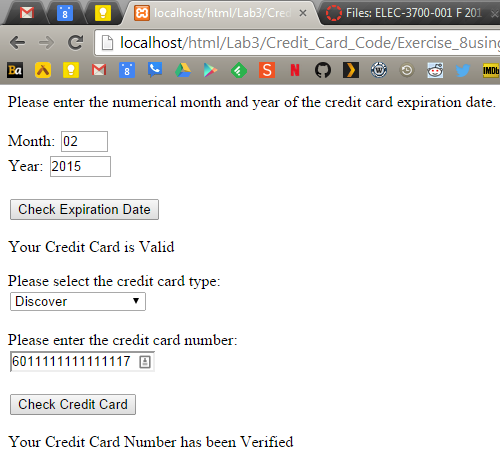


Figure 14. Verifying Discover

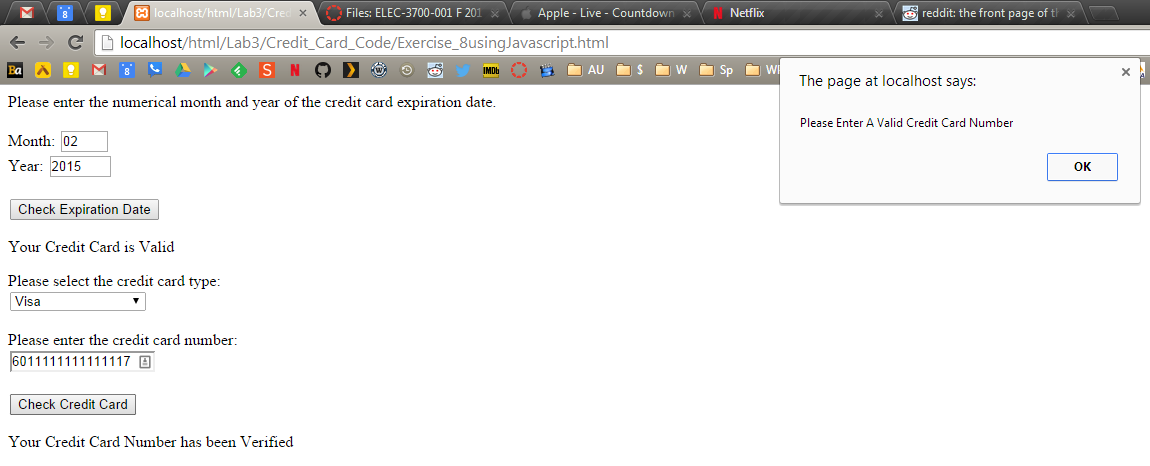


Figure 15. A bad Visa Credit Card number.

# Result

In this lab, we learned more about Regular Expressions, which I will always have to Google to create. We created some useful webpages to verify that user input is valid according to boundaries I set using the Regular Expressions, and also created a website to validate a credit card’s expiration date and number using the Luhn algorithm.

# Conclusion

This lab will serve as a great basis for the ecommerce project later in the semester. A good ecommerce website should be able to take and verify input from humans, such as for shipping purposes, and should be able to take in and verify credit card number and expirations dates. These are all things we learned in this lab, and are very useful.

# References

* Dr. Wu’s included slides and lab materials
* [stackoverflow.com/questions/5465375/javascript-date-regex-dd-mm-yyyy](http://stackoverflow.com/questions/5465375/javascript-date-regex-dd-mm-yyyy) for birthday Regular Expression