# COMP-206
# Software Systems
# **Assignment #4/5**

Due: December 5, 2014 at 23:30 on MyCourses

## Web Store in HTML, C, Perl & Python

This assignment is a team project. Your team can be 1, 2 or 3 people in size. I would suggest a size of three as being the ideal experience. I will be asking you to construct an online store. Your store can sell anything you like, be creative. One year, a team sold their friends, another year bobble heads of profs. Just make sure that what you pretend to sell is legal. Part of the goal of this assignment is to experience the reason for this course, which is, software systems in its full expression. In this light you will be combining the following things: HTML, CGI, C, Perl, Python, Shell, makefile, on a Linux web-server, and using a code repository in a team environment. This is what it is all about!

You are not an artist, but try to make your website look nice, even pretty.

Do not use web-authoring tools. Your only web-authoring tool is Vi. It is easy to identify a web page that was created using a web-authoring tool, especially when you look at the code. The TA will be looking at the code.

IMPORTANT: Since grading will be more difficult given that you are in a team, to help us, select a team leader. Your team leader will post in the discussion board in the "Register you Team Here" board the following: your team name, team member names (no ID numbers please) and what your store is selling. It is important to do this because the TA will be using this for grading. Everyone in your team submits the same material to myCourses for grading.

## Web-authoring Information

The following resources are available for you:

- McGill Web Server Info: http://socsinfo.cs.mcgill.ca/wiki/Webserver_FAQ (Public_html, Perl, Python, web services)
- CGI & Python: http://docs.python.org/2/howto/webservers.html
- CGI & C: http://www.cs.tut.fi/~jkorpela/forms/cgic.html#get
- Contacting help@cs.mcgill.ca for server help

## SETUP

To get this assignment functional you must first set-up your McGill web page directory. I suggest you try out the sample tutorials at:

http://socsinfo.cs.mcgill.ca/wiki/Webserver_FAQ - "Creating a personal web page".

Please do the following:

FIRST: Create a GIT repository. Each team member will have their own private GIT repository. The team leader's GIT and public_html directory will contain the official project (the TA will grade from the leader's public_html). Every team member will develop their portion of the project within their own personal public_html folder and GIT repository. Then they will give their completed code to the team leader. This can be done through standard file transfer or email, or you can use GIT push (but using GIT push is optional). All the code you develop: HTML, C, Perl and Python must be added to the leader's repository. The TA will verify if you used your GIT repositories as specified.

SECOND: Create a PUBLIC_HTML directory in the home directory of the team leader. This student will be hosting the web site. This directory needs to be chmod'ed as a public directory. All files within this directory that you want accessible by the outside world should also be chmod'ed as public. Test that this works by creating a simple index.html or default.htm file that says "HELLO WORLD". Then use the browser on the computer of another team member to see if that page can be viewed. If it shows up then you have set things up correctly.

THIRD: Develop code using the following technique: (1) assign work to each team member. (2) Team members develop their assigned task in their personal account privately (GIT your work). (3) Once the code is good move it to the team leader's account. (4) Integrate it into the official website (GIT your work). (5) Repeat. You may need to SSH files back and forth.

FOURTH: Create a text file called README.TXT and put your team member's names, ID numbers and a description of the work they were responsible for. Divide the work evenly among yourselves. Make sure you understand all parts of this assignment even though someone else might have done it. Make sure to include this file with your submission.

FIFTH: Enjoy this assignment, be creative. You are permitted to go beyond the scope of the assignment but you will ONLY be graded on what was asked for. The additional elements are for glory, not grades. **Do not replace requirements that give you grades for glory features because you will lose points**.

## The Web Site

Four web pages need to be created: a home page, a catalogue page, login page, and registration page. You need to determine what kind of store you will be presenting. Select your web site style and colours as a team. You can search the Internet for ideas to pattern your store on, but do not use their code. Do not go to template sights. For this assignment you must create everything by hand using Vi. You cannot use authoring tools. You will loose a lot of points if you try.

### HOME PAGE

The first page is the introduction page and should be programmed in HTML. It will introduce the store, provide an informative graphic and a "menu" to three web pages – you **cannot** use a web-authoring tool to write the code. The first menu link will be called HOME and will load (reload, in this case) the home page. The second link will be called CATALOGUE and will replace the home page by the catalogue page. The last menu link is called LOGIN and will display the login page in its own browser window or tab. In other words, two windows will be open at the same time: the home page (in this case) and the login page.

This three element "menu" must be present on all the web pages.

The home page must display the following information: Company Name, menu with links to the other pages, at least one graphic depicting what your site sells, and at least one paragraph describing the products your web store provides.

Using a <table> will be helpful to organize everything on this page.

Make the page attractive.

### WEB CATALOGUE

The second page will be the store's catalogue detailing the products the customer can purchase. It must be programmed using HTML and it will invoke a Python script. The page will have a title for each product, a graphic of each product, and a paragraph description of each product (minimum 3 items). Each product will have a check-box so the customer can indicate if they would like to purchase that item. The check-box is initialized as unchecked. Each item has a text-box for quantity initialized to zero. A submit-button at the bottom of the page will submit the CGI form to a Python script called Purchase.py (to be described later). This page should be attractive.

A user must be logged in for the submit button to work (to be described later).

**LOGIN PAGE**

The third page will be the login page. It will be a simple but good looking HTML and CGI page that will ask the user for a user name and password. The password text-box must not display what the user is typing. A submit-button will call the a.out file of a program called Login.c, a C program (to be described later).

This page will have a link to the registration page for users who do not have a user name and password.

**REGISTRATION PAGE**

The fourth page will be the registration page. It will be a simple but good looking HTML and CGI page that will ask the user for their full name, a user name, a password, and a confirmation password. The password text-boxs must not display what the user is typing. A submit-button will call a program called Register.pl, a Perl program (to be described later). This program will register the user.

## <u>The "Database" Files</u>

Your program will need the following CSV database files:

- Members.csv
    - Record structure for a user: user's full name, user name, and password.
    - **GLORY**: Ideally this file should be encrypted but it is not in our case.
- Inventory.csv
    - Record structure for an inventory item:
        - One word item name, quantity in stock, and unit price.
    - The catalogue should agree with this inventory file.
    - **GLORY**: Generate the catalogue web page from this inventory file.
- LoggedIn.csv
    - Tracks all users currently logged in.
    - Record structure: user name.
    - All users who have registered have their user name appended to this file

- **GLORY**: Have a logging out button and script to remove the user from this file.

## The Python Program

The Catalogue page is connected via CGI with a Python script that will handle the purchase. The submit button is connected to a Python script called Purchase.py. The catalogue page has a <input type=”hidden”> field that stores the user's user name if they have logged in, empty otherwise. Use this hidden field with step one of the algorithm below.

Purchase.py implements the following algorithm:

```
1. Verify if user is logged in. Compare the hidden field with the names in
   loggedin.csv.
   1.1 If not logged in then display error screen in HTML linking back to catalogue.
   1.2 If they are logged in then go to step 2.
2. Display a bill in HTML listing each item, quantity purchased and price. Include a
   total at the bottom.
   2.1 They are restricted to purchase only the amount actually in the inventory
   2.2 Subtract the amount purchased from the inventory
3. Provide a link back to the catalogue page and the home page at the end of the bill.
```

**GLORY**: Have your web site auto generate the catalogue page. List only those items that have a quantity in stock amount greater than zero. You will also need the inventory file to provide a description field with multiple sentences.

## The C Program

The login.c program lets the user log into the system. Once successfully logged in the catalogue page is automatically displayed but with an <input type=”hidden”> field containing the user's user name. This hidden field will be used by the Python program later on.

Login.c implements the following algorithm:

```
1. Verify that the user name and password are valid.
   1.1 If not valid then display an error HTML page with a link back to login and
       home.
   1.2 If the password is valid then go to step 2.

2. Append the user's user name into the loggedin.csv file.

3. Display the catalogue page but insert a hidden field and assign the user's user
   name to that field.
```

## The Perl Program

The Perl.pl program handles the request from the registration web page. This program appends the user information into the members.csv file if the user name does not already exist. If the user name exists then an error HTML page is displayed linking the user back to the Registration page and the Home page.

### WHAT TO HAND IN

Given that your web site is online, the TA will be able to run it using a normal web browser from anywhere. Do the following:

- Make sure the TA can see and run your web site online.
- ZIP everything (including the GIT files from all team members) and submit to My Courses.
- In addition, submit a simple HTML file with a single link to your operational web site. The TA will only need to click on it to get to your site.
- Also submit the README.TXT file.
- Your four HTML documents with supporting graphic files
- Your C, Perl and Python programs
- Your CSV files

## HOW YOU WILL BE GRADED

- Your program must run in order for it to be graded. If your program does not run the TA does not need to go any further and may give you zero.
- Your program must run on the Trottier computers. The TA will not modify your text files in any way to get them to run under Trottier's operating system.
- All grades are awarded proportionally. In other words, if a question is graded out of 4 points and you got 75% of the program running then you will get 3 out of 4, etc.
- Grading portions of your code that do not run (assuming that your program runs at least minimally) will be graded proportionally as to how correct it is.
- Make sure your code is easy to read since this may impact the quality of your grade.
- The assignment is worth a total of 20 points.
- Your grade is distributed as follows:

➔ This assignment is worth a total of 20 points

➔ Everyone in your group will receive the same grade regardless of what task they were assigned.

➔ Python – 3 points

➔ Perl – 3 points

➔ C – 3 points

➔ HTML – 3 points

➔ CGI – 3 points

➔ GIT – 3 points

➔ Following instructions – 2 points