FINAL GROUP PROJECT WRITEUP

**Required Installations:**

- PHP

- MySQLi

- Express

**Set Up:**

The website uses a MySQLi database to store user accounts. A MySQL user must be created and granted all permissions, with the username being “*connection*” and password “*student*”. However, this can be changed by going to the file named *“createAccount.php”* located in */var/www/html/php* and changing line 8 to the desired username and password. A database with name “*NewLeaf*” and a table called “*users*” must be created.

The website is set up so that the server also reads PHP that is embedded within .html pages. The default setup did not allow this. After searching through stackoverflow.com, we found that adding “﻿*AddType application/x-httpd-php .htm .html*” to the file located in */etc/apache2/sites-available/000-default.conf* fixed that issue.

To avoid going through the set up, we have included a .ova file that is fully configured and can run the website immediately. That file is: https://sfsu.app.box.com/s/eswx6te043wkama2i6yn52hmx19e9hzd

**Goal:**

Our initial goal was to create a website similar to what was submitted in our proposal. That included a carousel at the home page, among many other things. We had to deviate slightly at the beginning due to limitations of HTML and CSS alone. With JavaScript, we were able to get back on track and implemented a carousel display that can change images over an interval. Something that was implemented at the beginning was the use of “*<iframe>’s*” to handle the repetitive parts of the website, such as the navigation bar and the footer. Since Professor Bierman did not like <iframe>’s, we set it a goal to use PHP to handle it. Another mini-goal that was added later on was creating a store page that allows for an easy, hassle-free way to add items.

**Successes:**

As mentioned, we were able to implement a carousel-type image scrolling. It is set on a timer so it changes image every few seconds. We were also able to create a semi-dynamic store loading system. JavaScript was used to store our items into a global variable array. The array included item information such as name, description, category, image source, and price. A script then read off of the variable to display items into the store page. One of the more difficult areas was creating a script that can sort the items based on price or category type. Another success was implementing the backend. We used MySQL to store and read off of to handle registration and logins.

**Failures:**

One thing that we could have implemented had we had more time was to secure the passwords using hashcode and SALTing. Another thing that we wanted to do was to replace <iframes> with PHP—again, lack of time. The checking out process could also have been smoothened out. We also wanted to create a more complex user settings because, at the moment, it only allows for logging out and account deletion from the database. Finally, the codes could have been cleaned up and made more efficient. There are many files that may contain commented-out sections, unused variables, and redundant functions.