# PIC 40A: Homework 5 (due 11/16 at 10pm)

Like on homeworks 1, 3, and 4, it is important that you meet the following requirements.

- You must upload your files to **CCLE** before the deadline.
- You must upload your files to the **PIC server** in the appropriate directory before the deadline.
- Both submissions must be **identical** (down to the character).
  **Never** make changes to the PIC server submission after submitting your homework.
  (We can see when a file was last modified.)
- You must tell us (me and the grader) your **PIC username**.
- You must **validate** your HTML using `https://validator.w3.org/`.

In this assignment you will submit eight files. . .

1. `README.txt`. This will contain your PIC username.
2. `site_g.html` and `site_e.html`.
3. `welcome.php` and `phished.php`.
4. `phish.js`, `holiday1.html` and `holiday2.html`.

As mentioned above, you should submit all files to CCLE before the deadline.
You should also submit the files to the PIC server. Save them in the directory

`/net/laguna/???...???/your_username/public_html/HW5`

(in the folder `HW5` within `public_html`). We should all be able to view your live webpage at

`www.pic.ucla.edu/~your_username/HW5/site_g.html`

Now, I am just left to tell you what I want all of those files to achieve. Go over the page for that!

## So you know where we're heading...

In this homework, we're going to see how someone could phish users' passwords using PHP and JS.

Imagine that there is a website made by someone. We will refer to the creators of this website as `good`. On their website, users can log in and they can post comments about anything that they want to. `good` are not very careful and they allow their users to post raw HTML. On the login page, some recent users' comments are displayed.

In this situation someone can be `evil` and post a comment which includes mischievous anchor elements. We'll pretend that someone has posted a comment as niceGuy666. The comment reads "check out my holiday pictures!" The words "holiday" and "pictures" will link to pages which look like they have failed to load. "holiday" will direct to an error page that looks like it was produced by the PIC servers. "pictures" will direct to an error page that looks like it was produced by Google Chrome. Both of these error pages will load in a new tab, but here is the catch... They will cause the original login page to be redirected to one that looks the same but which functions differently. When a user enters their login information to this new page, `evil` will be able to acquire it.

## `site_g.html` and `site_e.html`

To a casual observer `site_g.html` and `site_e.html` will look exactly the same.

1. The tab will be called "Interesting".

   The main heading will say "Login page for site with interesting content".

2. There will be two sections:

   - one for logging in and
   - one showing "Recent posts by users".

3. The login section will use a `<form>` element with `method` set to `"POST"`.

   There will be two `<label>` elements and three `<input>` elements.

   The `<input>` elements will have their `type` attributes set to `"text"`, `"password"`, and `"submit"`, respectively.

4. The recent post by NiceGuy666 will link to `holiday1.html` and `holiday2.html`.

So what is the difference?

5. The web form in `site_g.html` will redirect to `welcome.php`.

   The web form in `site_e.html` will redirect to `phished.php`.

### Aside

Were this a real world example, `good` would likely have a PHP script run to show the recent users' posts. `evil` would then steal this as best it can. `evil` could not obtain the raw PHP, but it could obtain some HTML that looks realistic enough to fool a user.

### `welcome.php` and `phished.php`

`welcome.php` will look very similar to `site_g.html`. However, . . .

- The main heading will say "Welcome to site with interesting content".
- The login section will be replaced by a welcome section.

  It'll say "Welcome {}!!!" where {} is replace by the username that the user entered.

  **Note:** In this homework we won't bother checking the username and password are valid. This type of functionality will be addressed in a later assignment. In a real world scenario, this is something `good` should address.

`phished.php` will be even simpler.

- The tab will be called "Phished". The main heading will say "HAHAHA".
- There will be a paragraph saying "You just got phished!!! Your password is {}" where {} is replace by the password that the user entered.

  **Note:** In this homework we won't bother saving the username and password anywhere, but this type of functionality will be addressed in a later assignment. In a real world scenario, this is something `evil` could address in to order to be more evil.

### `phish.js`, `holiday1.html` and `holiday2.html`

- `phish.js` will be one line long. It'll use `window.opener.location` to redirect the opening tab. Both `holiday1.html` and `holiday2.html` should include this file.
- What about the rest of `holiday1.html` and `holiday2.html`? Well, first. . .

  **Note:** it is okay if `holiday1.html` and `holiday2.html` don't validate.

  This is because we're going to steal `holiday1.html` and `holiday2.html` from elsewhere.
- To get `holiday1.html` you can attempt to go to

  https://www.pic.ucla.edu/~mjandr/thisDoesNotExist

  You can then use the JavaScript console to ask for the `outerHTML` of the only `<html>` element, and copy and paste it into a new file.
- To get `holiday2.html` you can attempt to go to

  http://www.utternonsense.notawebsite.com/afterForwardSlash

  You can then use the JavaScript console to ask for the `outerHTML` of the only `<html>` element, and copy and paste it into a new file.

  The file is long, but you should be able to find `www.utternonsense.notawebsite.com` eight times and replace each occurrence accordingly: sometimes you'll want `www.pic.ucla.edu`; sometimes you'll want the entire path `https://www.pic.ucla.edu/~your_username/HW5/holiday2.html`.

  Also, you'll want to edit `reloadButtonClick` so that the `location` is updated to `https://www.pic.ucla.edu/~your_username/HW5/holiday2.html` regardless of how one specifies `url`.