Advanced PHP

WDV 441

Week 7

MVC Recap

- Recap of MVC Architecture
    - Consists of three parts
        - Model - manages the data
        - Controller - utilizes models to retrieve/edit data and sends the data needed for display on the view
        - View - receives the data from the controller and presents it to the user
    - When a PHP follows a common structure (PHP at the top, display at the bottom) it is easy to convert a page to MVC
    - We can setup a basic MVC structure by moving the HTML to a template and including it in the controller
    - Models and templates are usually stored outside of the public html in their own folders for security purposes and clear separation

WWW and state

- Unlike a desktop application or a phone app, WWW communications are by nature stateless
  - What does this mean?
  - Each page request knows nothing about previous requests made
  - Poses a problem for sites that need to track session state like a site with a login system
  - Information can be passed from page to page via the URL and GET parameters to maintain state but this is highly insecure
    - ie. http://mysite.com/index.php?user=gg
    - Why is this a bad idea?
- Along came cookies
  - Cookies allow the browser to store variables locally in files
  - While better than using the URL, still highly insecure
  - Malicious programs could read the files and glean information they could use to hijack your credentials and login as you
  - Cookies can "timeout" causing the data to be lost after a certain period of time

PHP Cookies

- PHP uses the function setcookie() to set a cookie
- Same as the header() function, setcookie() must be called before any text is output to the web browser
- When a page is requested, similar to $_GET and $_POST, $_COOKIE is available
- Utilize $_COOKIE to retrieve cookies that were previously stored
- I'm hungry let's take a bite!

Enter the Session

- Sessions utilize a cookie containing a session id on the client in combination with a file on the server to store state information
- Each request to the website will automatically send the session id triggering the server to reload the session from the session file for that page request
- In the event cookies are not supported, a web server can be configured to pass the session id through a GET parameter
- Sessions are the most secure of the ways to maintain state, but because it uses cookies it is still insecure
    - an attack called session hijacking steals the session id and can then use it to act as you on a website

PHP Sessions

- In order for PHP to initialize a session the function session_start() should be called
  - Like setcookie(), session_start() must be called before any text has been sent to the browser
- If a session is not already created it will be created and a cookie will be set to the session id
- session_start() will load any values that have been stored in the session file (on the server) and store them in the special $_SESSION array
- To save values to the session, store them to the $_SESSION array
- You can use the function session_id() to get the current session id
- Let's create a session

User Login

- Now that we have the ability to maintain a users session, we can investigate a user login
- User login is component of nearly every web application
- The basics needed for a user are
    - a unique identifier for the user (usually an autogenerated numeric field)
    - username (or email)
    - password
- Once the user logs in, the user id is usually stored to the session. This can be then used on every page to do things like check rights or utilize user options to control how the user interacts with the web application.

User Rights

- Most web applications have a need for more than one type of user
  - What types of users might a forum have and what rights might they have?
    - Guest
    - Contributor
    - Moderator
    - Administrator
- In order to be able to setup a system like this, we need a way to determine what the user can do
  - One of the most simple systems we can do is a user level system
    - Each user is assigned a specific level
    - Based on the level assigned, the user can access different areas of the system

Lab/Homework

- Using the news article class and pages as a starting point, create data table with the corresponding class and pages to add/edit users
- The user table should at least include the following fields
  - user_id
  - username
  - password
  - user_level
- Create a login page that will accept the username and password, verify the user credentials (create a function on the user class to do this), and start a session for the user once logged in
  - you will need to create a function on your user class to check login
  - the login page should also be MVC and will be very similar in structure to an edit page
  - after login you should store the user_id to the $_SESSION array

Supplemental Reading

- http://php.net/manual/en/ref.password.php

- https://www.w3schools.com/php/php_sessions.asp
- http://php.net/manual/en/ref.session.php

- https://www.w3schools.com/php/php_cookies.asp
- http://php.net/manual/en/features.cookies.php