CS 318 - Homework 9

Deadline:

Due by 11:59 pm on Wednesday, April 24, 2013

How to submit:

Submit your files for this homework using `~st10/318submit` on nrs-projects, with a homework number of `9`

Purpose

To read, think, and write about web application security, and to prepare for a slightly-larger application.

The Problems:

Problem 1:

Consider the OWASP XSS (Cross Site Scripting) Prevention Cheat Sheet:

https://www.owasp.org/index.php/XSS_Prevention_Cheat_Sheet

Read this cheat sheet, especially section 1.

Then, in a file `hw9-1.txt`:

- include your name
- write a paragraph (of at least five sentences) giving at least three important "take-aways" from this cheat sheet, and for each, why you chose it.

Note that these will be collected and posted to the course Moodle site.

Your resulting file `hw9-1.txt` is now ready to submit.

Problem 2:

Consider the "SQL Injection Attacks by Example" article:

http://unixwiz.net/techtips/sql-injection.html

Read this article, paying special attention to the "Mitigations" section near the end.

Then, in a file `hw9-2.txt`:

- include your name
- write a paragraph (of at least five sentences) giving at least three important "take-aways" from this article, and for each, why you chose it.

Note that these will be collected and posted to the course Moodle site.

Your resulting file `hw9-2.txt` is now ready to submit.

Problem 3:

Consider the Open Web Application Security Project (OWASP) site:

http://www.owasp.org/index.php/Main_Page

Start at this web page, and navigate from there, reading and investigating. Then, in a file `hw9-3.txt`:

- include your name
- write a paragraph (of at least five sentences) describing something you learned about web application security that you found interesting during your navigation/investigation from this site, and why you found it interesting. (This should be from somewhere within the site OTHER than Problem 1's XSS Prevention cheat sheet...!)
- include the URL within this site where you found the subject of your paragraph.

Note that these will be collected and posted to the course Moodle site.

Your resulting file `hw9-3.txt` is now ready to submit.

Problem 4:

You will be working on a slightly larger application across the next few homeworks. Because I want to allow you to choose between using Java servlets, JSP, and PHP for this, but I want you to start thinking about it now, you'll start by **JUST creating a finite state diagram for this application**, based on the following specifications. Then you'll implement it on **subsequent** homeworks.

You should create a finite state diagram for a 4-screen BOOKSTORE application as follows.

- First, read the following requirements carefully;
- Then, somehow create a PDF of a finite state diagram for this application (developed electronically, or hand-drawn and then scanned); submit it within a file with the name `bksales-fsm.pdf`

General Requirements

- you are required to use HTML5 on the client-side, and some JavaScript as well
- you are required to make appropriate use of CSS to maintain consistency between the different screens; make your screens as attractive and easy to use as possible
- you may use a combination of Java servlets, JSP, and/or PHP for the application tier-- your choice! But input validation is required, whichever combination you choose.
- you are required to use sessions to pass information between the four screens -- and you are required to invalidate/terminate your sessions appropriately
- you are required to make appropriate use of appropriate PL/SQL stored procedures/functions we have developed previously for this scenario (such as `sell_book`)
  - you are permitted to write and use additional PL/SQL stored procedures/stored functions as you wish.

- your **boilerplate** (that's just a term for "hard-coded", non-dynamic text) should be spelled correctly.
- currency formatting should line up appropriately through all 4 currency fields and should be to two fractional places

Screen 1:

- include an appropriate title for your bookstore
- include a way for the user to enter an Oracle username and password (the password field should be of type "password"!)
- include a submit button with the label "Log in"
- logging in should lead to SCREEN 2.

Screen 2:

- If the user types in an **invalid** username/password, **decide** which of these options you would prefer:
  - redirect back to SCREEN 1

    OR (slightly-more-informative, but requires an extra click by the users to retry):
  - show SCREEN 1A, which simply prints a "friendly" "that username/password combination didn't work" message of your choice with a "Back" button that takes you back to SCREEN 1 when it is pressed. This screen would be designed to be readable and user-friendly.

    OR (more advanced, I think):
  - redirect back to SCREEN 1 -- but that now also includes an eye-catching "friendly" "that username/password combination didn't work" message of your choice
- include an appropriate title for this 2nd screen for your bookstore.
- populate a `<select>` (drop-down box) element with ordered ISBN's from the bookstore database. Have it show at least 3 ISBN's at a time (that is, make use of the `<select>` element `size` attribute, which allows you to specify this).
- include a "Quantity sold" label and textfield, with contents initially 1.
- include a "Proceed" submit button, which leads to SCREEN 3, and an "Exit" submit button, which leads back to SCREEN 1.

Screen 3:

- what must be the case, if you have reached this screen appropriately? If you determine that any of that is NOT the case, decide which of these options you would prefer:
  - redirect back to SCREEN 1

    OR (slightly-more-informative, but requires an extra click by the users to retry):
  - show SCREEN 1A, which simply prints a "friendly" message of your choice describing the problem along with a "Back" button that takes you back to SCREEN 1 when it is pressed. This screen would be designed to be readable and user-friendly.

OR (more advanced, I think):

- redirect back to SCREEN 1 -- but that now also includes an eye-catching "friendly" message of your choice describing the problem

- include an appropriate title for this 3rd screen for your bookstore.
- include textfields and labels populated with the ISBN selected from SCREEN 2, its Publisher name, its Quantity sold (as selected from SCREEN 2), its Title, its Author, its Price, the computed Subtotal for a sale of this quantity, the computed Tax for a sale of this quantity (use a reasonable non-zero tax rate of your choice), and the computed Total for a sale of this quantity, including tax. These must be attractively formatted.
- use JavaScript to make sure only the Quantity field can be changed by the user at this point; if the user wants to change which book is being sold, he/she needs to use the Cancel button to return to SCREEN 2 (see below).
- the Price, Subtotal, Tax, and Total should all be displayed in Currency format to 2 fractional places
- include a Complete submit button that updates the database appropriately, using the `sell_book` PL/SQL stored function, and then leads to SCREEN 4.
- include a Cancel submit button that leads back to SCREEN 2 without updating the database.

Screen 4:

- what must be the case, if you have reached this screen appropriately? If you determine that any of that is NOT the case, decide which of these options you would prefer:
  - redirect back to SCREEN 1

    OR (slightly-more-informative, but requires an extra click by the users to retry):
  - show SCREEN 1A, which simply prints a "friendly" message of your choice describing the problem along with a "Back" button that takes you back to SCREEN 1 when it is pressed. This screen would be designed to be readable and user-friendly.

    OR (more advanced, I think):
  - redirect back to SCREEN 1 -- but that now also includes an eye-catching "friendly" message of your choice describing the problem
- include an appropriate title for this 4th screen for your bookstore.
- include something displaying a confirmation message of how many copies of the selected ISBN have been successfully sold.
- include an OK submit button that leads back to SCREEN 2.

Remember -- for THIS homework, **just** submit your resulting `bksales-fsm.pdf`.

Your resulting file `bksales-fsm.pdf` is now ready to submit.