



Seminar 2, A PHP Web Application

Internet Applications, ID1354

1 Goal

- Learn the PHP language.
- Learn to use PHP for server-side programming.

2 Grading

The grading is as follows:

0 points The mandatory tasks are accepted and you have passed the seminar.

1 point The mandatory tasks and one higher grade task are accepted. You have passed the seminar and have also gained one point to improve the final course grade, see course plan for details on final grade.

2 points The mandatory tasks and both higher grade tasks are accepted. You have passed the seminar and have also gained two points to improve the final course grade, see course plan for details on final grade.

To pass the LAB1 sub course you must pass all four seminars. If you fail this seminar you have to report it again at the end of the course, at the fifth seminar. You can also report higher grade tasks at the fifth seminar.

3 Auto-Generated Code and Copying

All HTML, CSS and PHP code must be well designed and you must be able to explain and motivate every single part. You are *not* allowed to copy entire files or classes from the sample chat application, even if you understand it and/or change it.

However, you are allowed to write code very similar to the chat application. You are also allowed to copy HTML and CSS from any web site and to use any web development tool, you do not have to write HTML and CSS by hand. In particular, you are encouraged to get inspiration from (or use) free design templates.

4 Mandatory Tasks

Tasks one to three must be solved and reported at the seminar. All those tasks involve storing data permanently on the server. One option for storing data is to use a text file, as in the sample chat application. Another option is to use a database, which is covered in lecture eight. Using a database is also an optional task for seminar three.

Task 1, Authentication

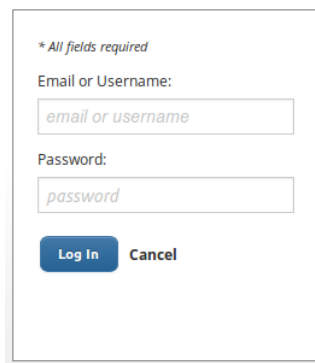


Figure 1: Login form.

Allow users to log in to the tasty recipes web site, by providing a form similar to Figure 1. The login form can either be placed on a new page, or be present on all pages, for example in the header. The login facility must match the style of the web site, this applies to font size, family and style; foreground and background color; mouse hovering and link behavior. None of these properties may have the default value just because it is the default. The tasty recipes site must still follow the five basic heuristics for user interface design mentioned in task three of seminar one.

Each user shall have username and password. This data shall be stored on the server, for example in a file or database. You are *not* required to let users register or update user information.

A typical mistake when writing HTML forms is to set the `action` attribute to a PHP function. *This is not allowed*, since it is just plain wrong. An HTTP request can never be directed to a PHP function, the value of the `action` attribute must be a URL, and execution will start from the first line in the PHP file at that URL.

- The report must show that it is possible to log in, and that the login part of the web site follows the five basic heuristics for user interface design mentioned in task three of seminar one.
- It must be shown in the report that values of action attributes are not PHP functions. You do not have to show all HTML forms, it is enough to refer to one particular form and explain what happens when it is submitted.
- The report must explain important parts of your code, and also include links to that code in your Git repository. Make sure your repository is public.

Task 2, Write Comments to Recipes

Users shall be able to write comments to recipes. Only authenticated users, those who have logged in as specified in task 1, can write comments (but all users can read comments). The tasty recipes site must still follow the five basic heuristics for user interface design mentioned in task three of seminar one. Comments shall be stored permanently on the server, for example in a file or database. Not only the comments shall be stored, but also information about who wrote the comment. The author's username shall be displayed together with the comment, for example as in Figure 2.



Figure 2: User comments to a recipe.

- The report must show that it is possible to write comments, and that the comment part of the web site follows the five basic heuristics for user interface design mentioned in task three of seminar one.
- The report must explain important parts of your code, and also include links to that code in your Git repository. Make sure your repository is public.

Task 3, Delete Comments to Recipes

Users shall be able to delete their own comments. This can be achieved for example by placing a `Delete` button beside each comment written by the currently logged in user. It is necessary to check, on the server, that the comment being deleted was written by the user who is logged in. If that is not the case, the comment can not be deleted.

- The report must show that it is possible to delete comments.
- The report must explain important parts of your code, and also include links to that code in your Git repository. Make sure your repository is public.



5 Optional Tasks

Optional Task 1, Register New Users

Allow users to register at tasty recipes. The data of a user account consists of username and password. Create a registration form where users can enter this data. The form shall be placed on a new page, with a style matching the other pages, this applies to font size, family and style; foreground and background color; mouse hovering and link behavior. The tasty recipes site must still follow the five basic heuristics for user interface design mentioned in task three of seminar one. The username and password of the newly created account shall be stored on the server, for example in a file or database, and used at login and for managing comments.

- The report must show that it is possible to create new user accounts, and that the registration part of the web site follows the five basic heuristics for user interface design mentioned in task three of seminar one.
- The report must explain important parts of your code, and also include links to that code in your Git repository. Make sure your repository is public.

Optional Task 2, XML

Use XML files to store all recipes on the web site. The XML shall be well formed and use the tag set defined in the mycookbok schema, see <http://www.mycookbook-android.com/files/mycookbook.xsd> and <http://mycookbook-android.com/site/my-cookbook-xml-schema/>. You can find a zip file with the same information on the *Assignments* page on the course web. Do not store any recipe data in HTML or PHP files. Use for example SimpleXML to parse the XML files, see <https://php.net/manual/en/simplexml.examples-basic.php>.

- The report must explain the content of an XML recipe file.
- The report must explain PHP code that parses an XML file and inserts data from the XML file into an HTML document.
- The report must explain important parts of your code, and also include links to that code in your Git repository. Make sure your repository is public.