Assignment 4—Databases

Overview

At this point, all of your webpages should exist in the form of servlets instead of static html pages. Those same servlets should also receive data from various forms, perform certain calculations on that data, and output the data “where it goes” on the webpage instead of a blank webpage. They should also implement sessions to handle login/logout and keep track of what form data the user has submitted. However, none of the data should currently persist through server restarts.

For assignment 4, you will take the next step by making your application's data **persistent**. This means that whenever the user buys stock or purchases items from your store, the application should remember it, even when the server is restarted. This should be done by using a database and not a flat file. To do this, you will probably want to remove (or significantly reduce use of) your hashmap and instead select from a database and insert into a database.

If you have not completed all of the requirements of assignment 3, you are encouraged to do so before beginning assignment 4, as assignment 4 directly builds upon assignment 3. As always, you may change your mind about the final project and choose a different option, but the assignment 3 content for the other option will need to be implemented before beginning assignment 4.

Assignment 4 Requirements

For assignment 4, the transactions that the user makes should be saved to a database and pulled from the database at the appropriate times. You must use MySQL as the database management system, since it is free, easier than some of the others, and is what I have installed to deploy your applications on. **No credit will be given for the use of flat files to enable the persistent functionality.** Installation instructions for how to install and configure MySQL are provided in the slides. It is recommended to install a full XAMPP installation, since this will probably be the quickest and easiest way to get MySQL installed. To configure the database, it is highly recommended that you use Phpmyadmin, which can be launched through XAMPP. Note that since I must be able to deploy your database, **submission instructions have changed**. Please see the submission instructions section for details.

Although an option on some previous assignments, please **do not use html 5 “required” attributes** for assignment 4. I must be able to tell that your server side validation is working.

AMMENDMENT
In assignment 4, please **do not use html 5 email attributes for input elements.** Please use text attributes for email addresses. I must be able to tell that your email validation is working. Also, please make sure that the email address is validated by the server (not with javascript or html) when accounts are created. Since there are varying degrees of validation, email addresses containing an @ sign and a

period will be considered valid email addresses.

1. User account registration page
In addition the the assignment 3 requirements, this page in assignment 4 contains the following new requirements:

When users are successfully created, they should be stored in the database. Also, please ensure email validation is performed, if not already.

2. Home/Login page
In addition the the assignment 3 requirements, this page in assignment 4 contains the following new requirements:

If a user has previously created an account and the server is restarted, the user should still be able to login through the login page. The best way to do this is to store your users in the database when the account is created. Then, this page can check if a user exists by checking the database instead of the hashmap. As mentioned previously, you will probably want to significantly reduce or eliminate use of a hashmap in this assignment.

3. Content Page 1
In addition the the assignment 3 requirements, this page in assignment 4 contains the following new requirements:

For the stock exchange option:
When a user successfully buys or sells stock, their stock holdings should be stored in a database instead of a hashmap.

For the online marketplace option:
This page requires no additional functionality beyond what was required in assignment 3. If the server restarts, it is ok if items in the cart disappear. If you choose to make the cart persistent, you may do so, but it is not required.

4. Content page 2.
In addition the the assignment 3 requirements, this page in assignment 4 contains the following new requirements:

For the stock exchange option:
This page should display accurate information about the currently logged in user. This information should **not** be reset when the server restarts and should be pulled from a database. The information it should display is the same as in assignment 3. This information includes the user's: username, first name, last name, email address, and all other information created at account creation time except the password. It will also display the current stocks the user holds, how many shares of each they hold, and the total value of their holdings. Likewise, it will show the remaining balance of their account. Also, for assignment 4, when the add money button is clicked, the funds should be permanently added to the user's account and should not be reset on server restarts. For assignment 4, please do not generate a separate webpage when the user adds money to his/her account. Instead, please

display the success message on the same webpage.

For the online marketplace option:
The cart does not need to be persistent across reboots. However, on this page, when the checkout button is clicked and the user has enough money, the transaction should be saved in the database. Also, when the add money button is clicked, it should permanently add the entered funds to the user's current account balance. The old assignment 3 requirements involving success and failure messages still apply.

5. Content page 3. (if online marketplace option is chosen)
If you have selected the online marketplace option you will need to make your content page 3 contain an accurate order history of all the logged in user's orders. This means that this page should show each order, the items and quantity of items in each order, and the total cost of each item. It
will also show the total cost of each order. The page will also distinguish between each order and display a grand total for all orders combined (this is the total amount of money the user has spent on the site). Also, since it was not required in assignment 3, in assignment 4 If the user is not logged in, the page should redirect (not forward) to the login page. If the user is logged in, the page should display a message somewhere saying “Hi (logged in user)” where (logged in user) is the logged in user's first name.

Independent Projects

If you proposed an independent project and it was approved, in this assignment, you must make all of your application's data persistent. This means it must persist even if the server is restarted. It must do this by using a MySQL Database. Also, note the modified submission instructions. If you have questions about what data should and should not be persistent, please contact me at least one week before the due date. You do not need to do anything involving web services or third party api's for this assignment.

Additional Requirements for All Assignments

The site should use css to implement some sort of simple color scheme and be reasonably aesthetically pleasing. (Basically it should not look 'tacky').

All pages should be linked to all the other pages. Links should be in a consistent position. They should not move or break from page to page.

Submission Instructions (NOTE: There is an added requirement)

New in Assignment 4
When you submit this assignment, in addition to the submission url and zip file, you must submit an sql script file which sets up your database and tables on my system. This is because you cannot “upload” a

database into moodle. This script is a file with an sql extension and several sql statements which create a database, create a table or several tables, create a user, and assign permissions to that user. I have created a template for you to use and uploaded that template into moodle. You should be able to fill out the template with the necessary information for your database. **IMPORTANT: Please verify your sql script works before turning in your assignment. If your database does not setup properly, the assignment will receive a 0.** You can test this by deleting your existing database (preferably after backing it up) and running the statements contained in your .sql file. If your app works properly after this, your sql script is good to go.

If you are using netbeans, please make sure that you correctly **EXPORT TO ZIP** within netbeans as shown in class. This means, within netbeans, click "file" at the top left, then navigate to "Export Project" and choose "To Zip..." This will open a dialog box. Select your project and click the "Export" button. Do not simply zip the directory itself with windows or your project will not load into my netbeans installation correctly. **If it does not load into my netbeans installation correctly, you will get a 0 on your assignment.** To test whether or not you are doing it correctly, please backup your project, export it from netbeans, delete it from netbeans, and import your exported copy. As with the last assignment, you do not have to adhere to any naming convention, but if you would like to, you may name your project NetbeansLastname where Lastname is your last name. Please upload this zip file into moodle.

If you are using eclipse, the instructions have also changed. The zip files do not load properly into netbeans so I will be grading your applications with a separate eclipse installation. Eclipse has the ability to easily export war files with the source code. **Please export a war file of your project** by right clicking your project within eclipse and navigating to the option called "Export." Choose "war file" and a dialog will appear. Check the box for "export source files" and click finish. Save your file somewhere. Rename your file "EELastname" where Lastname is your last name. Please upload this file into moodle. Please adhere to this naming scheme so I know to use eclipse to load your application. Do not simply zip the directory itself with windows or your project will not load into my eclipse installation correctly. **If it does not load into my eclipse installation correctly, you will get a 0 on your assignment.** To test whether or not you are doing it correctly, please backup your project, export it from eclipse, delete it from eclipse, and import your exported copy.

For both netbeans and eclipse, provide a url for your application's home page or login page in the comments section in moodle. Unlike 2300, this will be a local url. It will probably look like one of the following

http://127.0.0.1/yourpackage/yourservlet

or

http://localhost/yourpackage/yourservlet

but definitely not

C://users/name/workspace/somefile.html

Please do not attempt to use notepad++ or dreamweaver for this assignment. They both work great for simple web development but do not support dynamic web development at all.

Grading Rubric

10 points -All site content present and site links working and in consistent position. Css styles implemented. Site has color scheme and has a good look and feel. All pages are generated from servlets instead of existing as static html files. Servlets do not crash when some form data is not provided to them, and instead display the appropriate messages.
10 points- Server side account validation and creation works, even if users are not saved persistently. Login page works correctly. Redirection when users are not logged in works correctly.
10 points- All additional assignment 3 functionality working. This includes purchasing/selling of stocks for stock sites and carts for marketplace sites.
20 points- User accounts correctly being stored in database when account is created. Login works after server restart.
10 points- Correct user balance and user account creation data persist through server reboots. This information is displayed on the correct page when the user logs in after a server reboot.
40 points- Other new requirements met. This includes, among other things, correct displaying of persistent stock holdings for project option 1 or persistent order history for project option 2. This information must persist through server reboots and be stored in a database.

Total 100 points

Submission Checklist

Don't forget: For assignment 4 you need to turn in these three things:

1. Submission link
2. Sql script
3. The correctly exported zip file.