Assignment 3—Servlets Part 2 and Sessions

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 that data onto a blank webpage. In this assignment, you will take the next step by outputting the data "where it goes" on the webpage instead of a blank webpage. You will also implement sessions to handle login/logout and keep track of what form data the user has submitted. If you have not completed all of the requirements of assignment 2, you are encouraged to do so before beginning assignment 3, as assignment 3 directly builds upon assignment 2. As always, you may change your mind about the final project and choose a different option, but the assignment 2 content for the other option will need to be implemented before beginning assignment 3.

Assignment 3 Requirements

For assignment 3, all pages (except your css and images and javascript) of your project should already exist as servlets and generated dynamically. This means that they will no longer exist as html files, and will instead be "spit out" line by line by a servlet. In this assignment instead of your servlets spitting out the data they have received on a blank page, they will output the data "where it should go." Details are provided below. **Note that all of the users and their date will be deleted when you restart your tomcat server** if they are stored in the session. This is normal and expected for this assignment.

AMMENDMENT:
Although an option on previous assignments, please **do not use html 5 "required" attributes** for assignment 3. I must be able to tell that your server side validation is working.

1. User account registration page

The user account creation page should output the registration form if the user is not logged in. It should also process data sent by the registration form. If some of the data in the form is missing or improperly formatted, it should display a warning message showing what information is missing or incorrect and should not create an account. Do not worry about conflicting usernames for this assignment. The message can be in the form of an alert or text on the page. It is recommended to output text directly on the page rather than through an alert, since it will probably be easier. If the data passes validation, the servlet should generate a success message and create the user account. The account should be created with a $0 balance. If a user visits the page and they are already logged in, the page should not display any form fields and should instead display a "Thank you for registering" message. NOTE: The validation MUST be performed with java, not javascript (javascript validation can be bypassed by anyone). No credit will be given if validation is performed with javascript. Obviously, if an alert is used to display the message, the alert will be created with javascript, but the validation should be done with java.

2. Home/Login page

If no user is logged in, the login page should display the login form. If a user is logged in, the page should not display a login form and should display somewhere a message saying “Hi (logged in user)” with the actual first name of the logged in user in place of (logged in user). It should also display a link or button called “logout” which submits to the home/login page. When the logout button or link is clicked, the login page should logout the current user and display the login form, just as if a user was not logged in.

3. Content Page 1

For the stock exchange option:

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 and the page should display the normal page with the form fields and buttons. When the user clicks a “buy” button beside a stock and enters a quantity in the quantity field, if the user has enough money, the servlet should add the quantity of shares of that stock to the user's session and subtract the cost from the users balance and generate the usual webpage (not a blank page) with a success message stating the name of the stock, the quantity being purchased, the total purchase price, and the user's remaining balance. If the user does not have enough funds, the servlet should display the usual page with a message that says “insufficient funds” and should not subtract any money from the user's balance. When the user clicks a “sell” button beside a stock and enters a quantity in the quantity field, if the user has enough shares, the servlet should subtract the quantity of shares of that stock from the user's session and add the value to the users balance and generate the usual webpage (not a blank page) with a success message stating the name of the stock, the quantity being sold, the total sale price, and the user's new balance. If the user does not have enough shares, the servlet should display the usual page with a message that says “insufficient shares” and should not add any money to the user's balance. If a quantity field is not provided during a buy or sell transaction, the servlet should generate the usual webpage with a message that says “No quantity entered.”

For the online marketplace option:

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 and the page should display the normal page with the form fields and buttons. When “add to cart” is clicked for an item, the servlet should add the quantity of the item chosen item to the user's session. The page should also display a success message saying “Items successfully added to cart” You do not need to print any details in this success message, just the generic success message.

4. Content page 2.

For the stock exchange option:

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 and the page should display the normal page with the form fields and buttons. If the user is logged in, this page should list out the accurate current user information. This means it will display the logged in 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. It should already have a

field for adding money to their account and a button called “add money” that adds the money to their account. For the purposes of this project, we will pretend that whatever amount of money the user places into the field is given to them for free when they click the add money button. When the user clicks the “add money” button, the servlet should generate the usual webpage with a message that says that says “added funds” and the total amount of money the user entered into the field. It should also add this money to the user's current account balance. If no amount is entered, it should not crash and should display a message stating “no amount entered.”

For the online marketplace option:

In this assignment, we will make the cart “work.” 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. The page should also show the user's current cart (which will be empty if they have no items). It will show the name of each item in the cart, the quantity of that item in the cart, and the total cost of the item (item price x quantity of item). It will also show a grand total. It should already contain two buttons, one called “Clear Cart” and one called “Checkout.” If the “Clear Cart” option is chosen, the servlet will show an empty cart and a success message saying “Cart Cleared.” If the checkout option is chosen and the user has enough money in their account, an empty cart will be shown along with a success message and the new balance. If the user does not have enough money in their account, the cart will not be cleared and an error message will be displayed stating that the user does not have enough money. Additionally, the page should already contain a field for adding money to their account and a button called “Add Money” that adds the money to their account. For the purposes of this project, we will pretend that whatever amount of money the user places into the field is given to them for free when they click the add money button. When the user clicks the “add money” button, the servlet should show the current cart with a message that says that says “added funds” and the total amount of money the user entered into the field. It should also add this money to the user's current account balance. If no amount is entered, it should not crash and should display a message stating “no amount entered.”

5. Content page 3. (if online marketplace option is chosen)

If you have selected the online marketplace option and have a content page 3, no additional content is required for content page 3 in this assignment. It should have all of its fields and data (from assignment 1) and should exist as a servlet instead of an html file (from assignment 2), but no additional work is needed in assignment 3. Please keep the page though, since we will make it work in assignment 4.

Independent Projects

If you proposed an independent project and it was approved, in this assignment, you must fully implement your application functionality (except for stuff requiring web services or third party api's). As with assignment 2, servlets must receive all data from your forms and process that data as necessary. However, they should output this data where necessary on their web pages. The application must also have fully working login, logout, and registration (as detailed for the two main projects above), and all users who visit pages and are not logged in should be redirected to the login page. Each page should also display a message that says “Hi (logged in user)” where (logged in user) is the logged in user's first name. No persistence or database connectivity is needed in this assignment.

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

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

5 points -All site content present and site links working and in consistent position. Submission file is named correctly.
5 points- Css styles implemented. Site has color scheme and has a good look and feel.
5 points- All pages are generated from servlets instead of existing as static html files.
10 points- Servlets do not crash when some form data is not provided to them, and instead display the appropriate messages.
20 points- Server side account validation and creation works
20 points- Login page works correctly. Redirection when users are not logged in works correctly.
35 points- All additional functionality working. This includes purchasing/selling of stocks for stock sites and carts for marketplace sites.

Total 100 points