Remaining questions:
1. when to delete outdated data in db
2. change the db name to test 500
3. user id invalid case
4. none visible case

COMP3322 Modern Technologies on World Wide Web

Assignment Three

Total 16 points

Deadline: 23:59 Nov 21, 2023

Overview

You are tasked with developing a mock-up financial website that showcases a dashboard of financial data. This website should provide users with the ability to personalize and customize the organization of the dashboard. To achieve this, you are required to utilize PHP for server-side scripting, along with a MySQL database to store and manage the user's personalized settings.

Objectives

1. A learning activity to support ILO 1 and ILO 2.
2. To practice how to use HTML5 drag-and-drop features, PHP, cookies, and MySQL to create a mock-up financial website that supports a personalized dashboard.

Specification

You develop the application using the course's LAMP docker containers. (Note: another platform you can use for the development is the department's i7 Web server and Sophia MySQL server.)

You will develop three programs – index.php, index.css, and index.js. The file index.php contains PHP code for generating the HTML content that will be sent to the client. The file index.css contains all CSS rules for this application. The file index.js contains JavaScript code for handling all client-side dynamic actions, such as drag-and-drop and support customization. The files should be placed in the public_html folder of the web server docker container with the following **directory structure**.

```
public_html
├── images
│   ├── Convert-Currency.png
│   ├── Crypto.png
│   ├── FTSE100.png
│   ├── HSI.png
│   ├── SP500.png
│   ├── USD-HKD.png
│   ├── ex_rate.png
│   ├── eye-close.png
│   ├── eye-open.png
│   └── nasdaq.png
├── index.css
├── index.js
└── index.php
```

You can download a copy of the images.zip file from the course's Moodle site. This zip file contains all the images; some images are for the dashboard and two images are for the JavaScript dynamic feature. Uncompress the zip file and place the images folder directly under the public_html folder.

dashboard.txt

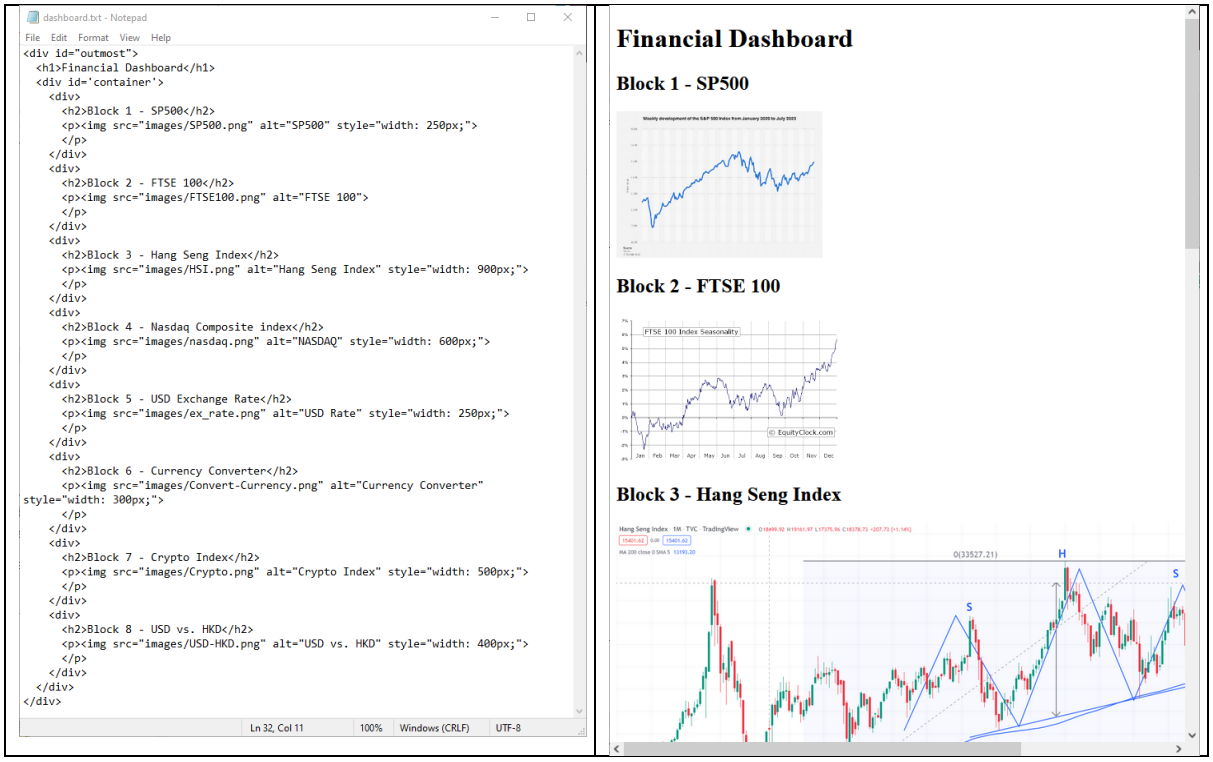
Our mock-up financial dashboard contains of eight financial data blocks, each contains an H2 element followed by an image. The initial order of these financial blocks is determined by the H2

elements, with Block 1 being the first block and Block 8 being the last block. Each block contains a unique image, and the display dimension of each image **is fixed**.

This financial dashboard contains an H1 title element and the whole dashboard is contained inside a DIV element.

As it is a requirement to set all the blocks in this order initially and ensure all images are displayed according to their defined settings, we have captured these specifications in the dashboard.txt file. This ensures that everyone to get the same HTML setting. You can download a copy of this file via the course's Moodle site.

Below are the screenshots of the dashboard.txt file (left) and the browser's output when rendering the HTML content of this dashboard.txt file (right).



index.php

Implement the index.php program to handle the GET /index.php and PUT /index.php requests.

GET Method

This method is for users to access the financial dashboard page. This page makes use of cookies to identify users and associates the users' personalized settings (which are stored in the backend database).

Here are the services provided by the program:

- If a client hasn't visited the page before, the program will send the dashboard content with the default setting together with a unique user ID (via the cookie) to the client.
- If a GET request carries a valid user ID, the program will check whether this client has a custom display setting. When this client has a custom setting, the program will rearrange the content as specified by the user's custom setting before sending the content to the client.

handle the case of all blocks invisible

When this client does not have a custom setting, the program will send the content with the default setting to the client.
[**Note**: The server always sends all eight financial blocks to the client; some financial blocks are set to 'visible' and are ordered according to the user's stored preference and the remaining financial blocks are set to 'invisible'. The system needs all financial blocks for users to make changes to their preferences.]

when will this case happen, how to test it

- If a GET request carries a user ID not in the database (i.e., invalid ID), the program will send the content with the default setting together with a new user ID to the client.
- The program should set the expiration duration for a new user ID cookie to **5 minutes**.

The response generated by PHP should consist of the HTML code only. All styling rules, except for the financial images' settings, should be placed in the index.css file, while all JavaScript code should be placed in the index.js file. These files are linked to the HTML code, and the browser downloads them when rendering the page.

Here is an example look of the page without personalized settings. In addition to the HTML code in the dashboard.txt file, we need to add a "button" for the user to enable the customization feature.

PUT Method AJAX

This method is for the client-side to send the user's personalized display setting to the server for storage. The client sends an asynchronous message and uses cookies to carry the user's preferences to the server. The cookie(s) is/are generated by JavaScript at the client end.

As the system allows users to select which financial blocks to display and in which order, their preferences sent to the server should contain this information. To reduce server loading, other information can also be sent. For example, one cookie can carry the 'visible' set, while another can carry the 'invisible' set, or both can be combined into a single cookie. It is worth noting that the invisible set is an optional feature, as the server can derive the information.

Here are the services provided by the program: visible set and the order of visible blocks

- Check whether the PUT request carries a user ID and the cookie(s) contain the user's preference; if not, the program should reject the request and send a response with the code 400 to the client.
- If the user ID is valid, the program should store the user's preference in the database and send a 200 response.
- If the user ID is not valid (i.e., not in the database), the program should create a new user ID, store the user's preference (with the new ID) in the database, and send a 200 response that carries the new user ID to the client.
- If the server encounters an error in storing the data in the database, the program should send a response with the code 500.
- For all cases, if the requests carry cookie(s) that contain the user's preference, the response traffic should inform the client to remove the cookie(s), i.e., after the event, the cookie file at the client side should not have the user's preference cookie(s).

User database

The program stores all users' personalized settings in the database. Here is an example database table for storing the information. [Note: you can have your own database design, and this is for your reference.]

uid (key)	***visible***	***hidden***	***timestamp***
idd46228	"blk5", "blk6", "blk2", "blk4", "blk1"	"blk3", "blk7", "blk8"	1695718226
idd60770	"blk3", "blk5", "blk1", "blk6"	"blk2", "blk4", "blk7", "blk8"	1695718316

In this design, the "visible" field is for storing the IDs of all visible financial blocks together with their order of appearance; the "hidden" field is for keeping track of which blocks are hidden. The timestamp field stores the expiratory time of the cookie measured in the number of seconds since the Unix Epoch.

Data Housekeeping

The program should check the database to clear all outdated user records. When a user's cookie has expired, the corresponding record should be removed.

The program should support these operations in managing the database:

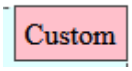
- Add a new user record for a new user ID
- Remove an outdated record when the user ID expired

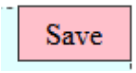
- Update a user record when the user submits a new personalized setting
- Retrieve a user record according to the user ID

index.js

This is the client-side program that runs on the browser to support user customization of the web page. Here are the basic services provided by this JS file.

1. The “Custom”/”Save” button
   This is a toggle button. The user clicks on the “Custom” button, it enables customization. The name of the button will change from “Custom” to “Save”. Once the users finish customizing, they can click on the “Save” button to save their personal settings to the server and disable further customization.

   When the customization effect is enabled, the user can set a block to become visible or hidden and can rearrange the financial blocks by drag-and-drop.
2. Set the visibility setting of each financial block
   After enabling customization, all eight financial blocks should be displayed on the screen no matter whether they are in a visible or hidden state. There is a red border box (“1px dashed red”) that wraps around the block.
   By default, all blocks are initially in the visible state, unless the user turns a block to the hidden state. When a block is in the visible state, it displays an open-eye icon in the top right corner. Upon clicking the open-eye icon, the program changes the state of the block from visible to hidden, updates the icon to a close-eye icon, and moves the block to become the last block on the page.

   The block is in a visible state.

   The block is in hidden state.

   When the user clicks the close-eye icon, the program changes the state of the block from hidden to visible, changes the icon to an open-eye icon, and moves the block to become the first block on the page.
3. Allow moving a financial block by Drag-and-Drop operation
   A user can rearrange the order of both visible and hidden financial blocks by dragging and dropping them. When a financial block is dragged over another financial block, it becomes the next sibling of the target financial block. Note that the entire financial block is moved during this process. For example, here are screen captures demonstrating the before-and-after states of the drag-and-drop feature.

Before dragging block 7 over block 8

After dropping block 7 over block 8

To enhance the user experience, the border of the target droppable block should change to "5px solid pink". This change indicates that this droppable block allows a drop event **at that location**. After the drop event or when the dragged block has left the droppable block, the program should remove this effect and reset the border box to "1px dashed red".

When dragging block 6 over a position on block 8 that accepts the drop event, the border of block 8 changes the color to "5px solid pink".

After the drop event, the border of block 8 resumes to normal color.

4. Save the user's preferences and disable customization
   When the user clicks the "Save" button, the program should perform the following tasks.
   - Collect the current set of financial blocks that are in the visible state and their order, then send the information to the server by using the AJAX technique (e.g., fetch, XHR, $.ajax()). The program should generate some new cookie(s) for carrying the user's preferences to the server. Display the returned status code to the console log for debugging purposes.

     user_id cookie expire first and then click save btn
   - Disable the drag-and-drop operation.
   - Remove the eye icons and the border boxes of all financial blocks.
   - Set all financial blocks that are in the hidden state to become invisible.
   - Change the label of the button to "Custom".

For this assignment, you can use vanilla JavaScript and/or jQuery library for the implementation; no other external libraries are allowed.

index.css

You should place all the CSS rules in this file. Here are the suggested settings for some elements on the page.

- For the DIV element with the id="outmost", set the width to 1000px and margin to "1rem auto 0".
- For the DIV element with the id="container", set the width to 100%, padding to "30px 10px", and box-sizing to border-box. Select a background color and a border setting of your choice. Set this container to be a Flexbox container that supports wrapping, justifies the content by space between, and aligns the items by flex-start.

For the other elements on the page, you should make your judgment in setting appropriate CSS rules for them.

Resources

You are provided with the following files.

- `dashboard.txt` – this text file contains the HTML code for this webpage.
- `images.zip` – this zip file contains all image files used by the program.

Testing platform

We shall run the PHP program in the LAMP container set and use curl and Firefox/Chrome to test the program.

Submission

Please finish this assignment before Tuesday November 21 23:59. Submit the following files:

1. index.php
2. index.js
3. index.css
4. Export a copy of the table that stores the users' preferences from your MySQL database and submit it to Moodle. [Note: the system usually uses the table name as the filename with the .sql extension when exporting the table, e.g., cookie.sql]

Grading Policy

Points	Criteria
7.0	index.php GET method (3.0) ▪ correctly use cookie to identify a user ▪ correctly return the dashboard data according to the user's history and preferences PUT method (3.0) ▪ use cookie(s) to carry a user's preference ▪ remove the cookie(s) after the event ▪ correctly handle the PUT request if provided with the correct data ▪ return an appropriate response for all situations Data housekeeping (1.0) ▪ remove outdated records
8.0	index.js ▪ implement the "Custom"/"Save" button ▪ only display those financial blocks that are in the visible state and order the blocks according to the user's preferences

	▪ display all financial blocks after enabling customization ▪ add the correct eye icon to indicate the visibility state of each block ▪ use the icon to support switching between states and locations ▪ correctly implement the drag-and-drop operation ▪ adjust the border of the droppable block during the drag-and-drop event ▪ use the AJAX technique to send a PUT request to the server to store the user's preference ▪ correctly switching back to normal mode
1.0	index.css ▪ implement the flexbox setting to the DIV element with id="container" ▪ apply basic styling to elements
-4.0	Should not change the structure and setting of the eight financial blocks
-4.0	Using any external libraries other than jQuery
-4.0	Did not provide the SQL file in the submission

Plagiarism

Plagiarism is a very serious academic offence. Students should understand what constitutes plagiarism, the consequences of committing an offence of plagiarism, and how to avoid it. ***Please note that we may request you to explain to us how your program is functioning as well as we may also make use of software tools to detect software plagiarism.***