Assignment 2

Server-side programming and AJAX

Submit a single zip file called **assignment2.zip**.
This assignment has 100 marks.
You should read the marking scheme posted on cuLearn for details.

Assignment Background

In this assignment, you will develop a server capable of serving the order form resources from assignment #1. The server will also be responsible for tracking some sales data for each restaurant and providing that sales data in HTML format when requested. All order data that your server stores for this assignment may be stored in RAM. That is, there is no need for persistence. If your server is restarted, it is acceptable to start from an empty set of orders. **The only external modules you may use for this assignment are template engine modules.** For the parts that require a template engine, you may use any available template engine you desire.

Before starting your design for the assignment, you are encouraged to read through the entire specification. It is possible that later problems will inform your design decisions and, by preparing in advance, you can avoid having to significantly refactor your code as you progress through the assignment.

To begin this assignment, download the `assignment2-basecode.zip` file from cuLearn. This zip file will contain a basic solution to assignment #1, as well as a `restaurants` directory containing three .json files. Each of these files contains the restaurant data for one restaurant in JSON format. Note that JSON within a file can be easily read into your Node.js server and converted to a Javascript object using the `require` directive.

The first part of this assignment will involve serving resources like those created for assignment #1. You may use your own assignment #1 solution (recommended if it meets the requirements), start from the files provided in the zip, or write a new solution that meets the minimum requirements listed in the problem description.

Server Load (10 marks)

When your server loads, it must first read and store all relevant information contained in the files within the `restaurants` directory. Your code should be written in such a way that new files may be added to the directory containing additional restaurants' information without requiring changes to your code. Your server should not start until this initialization process has completed.

Home Page and Header (15 marks)

Your server must serve a home page in response to GET requests for the URL "/". At minimum, this page is required to display a basic welcome message, though you are free to add extra content. In addition, this page and all other pages within your application must display a header containing links to the following resources:

1. The home page
2. The order form
3. The restaurant stats page

Order Form (30 marks)

When requested (e.g., through clicking the link in the header), your server must respond with the resources required for the order form to function. As mentioned above, you can modify your own resources from assignment #1, use the provided resources, or write new resources. The requirements for the order form are listed below:

1. When the page loads, the list of restaurants used to populate the drop-down menu must be requested from the server.
2. When a restaurant is selected from the drop-down menu or the drop-down menu selection changes, the new restaurant data must be requested from the server. The previously selected restaurant's data must be removed.
3. The page must provide, at minimum: a list of all menu items for the currently selected restaurant, the price of each of those items, a way to add items to the order, and a summary of the order contents/price. If the page does not support this, there will be no way to test much of the server functionality and your grade will reflect that.
4. When the Submit Order button is clicked, the order must be sent to the server (i.e., using an XMLHttpRequest). Your server must process the received order data and update any local state required to implement the rest of the assignment requirements (e.g., updating restaurant order totals, etc.). Once the request has been processed successfully by the server, and a response has been received, the client's page must be reset to its initial state.

Restaurant Statistics Page (30 marks)

When requested (e.g., through clicking the link in the header), your server must respond with a page containing the following information for each restaurant:

1. Total number of orders received
2. Average order total (subtotal + 10% tax + delivery fee).
3. Name of the most popular item, or the name of one of the most popular if there is a tie. The most popular item is the one that has sold the most units (multiple units in an order should be counted multiple times).

You must use a template engine for this page. Solutions that do not make use of a template engine will receive no marks.

Code Quality and Documentation (15 marks)

Your code should be well-written and easy to understand. This includes providing clear documentation explaining the purpose and function of pieces of your code. You should use good variable/function names that make your code easier to read. You should do your best to avoid unnecessary computation and ensure that your code runs smoothly throughout operation. **You should also include a README.txt file that explains any design decisions that you made, precise instructions detailing how to run your server, as well as any additional instructions that may be helpful to the TA.**

Recap

Your zip file should contain all the resources required for your assignment to run.
Submit your **assignment2.zip** file to cuLearn.
Make sure you download the zip after submitting and verify the file contents.
