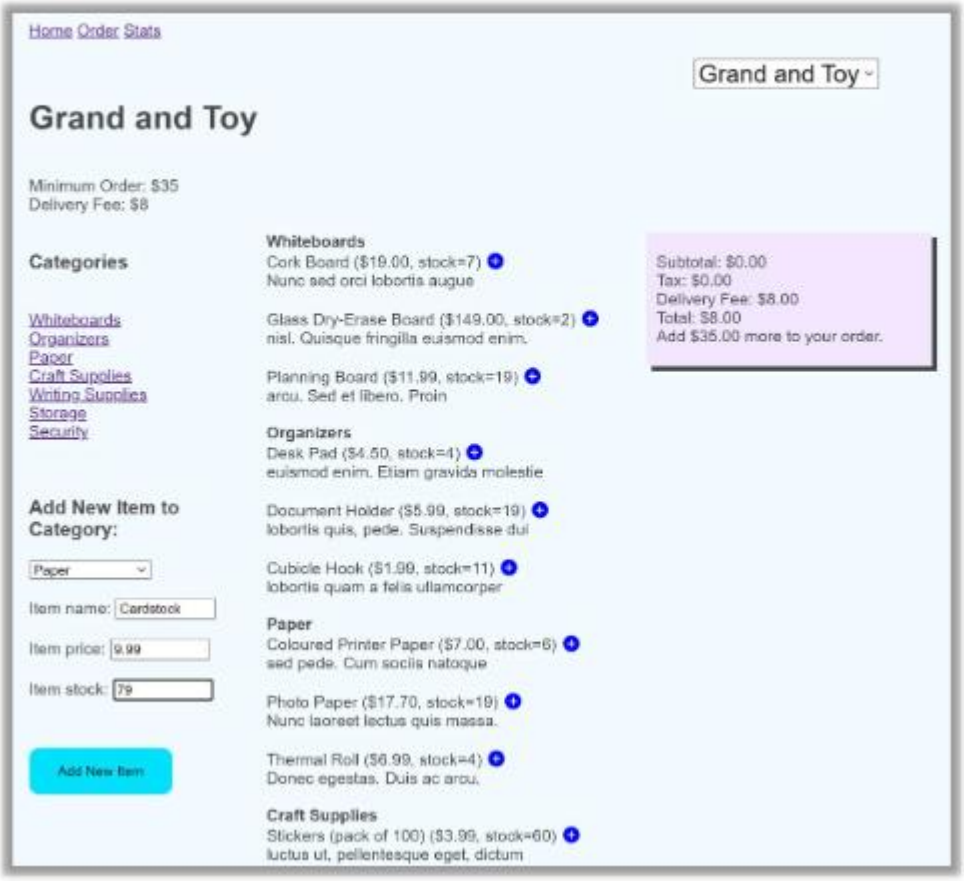
This assignment has 100 marks.

Web Page Sample



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 vendor 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.

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.

Provided files

To begin this assignment, download the **a2-base-code.zip** file from Brightspace, that contains 9 files:

- **orderform.html**
- **styles.css**
- **client.js**
- **server.js**
- **add.png**
- **remove.png**
- **vendors/grand.json**
- **vendors/indigo.json**
- **vendors/staples.json**

This zip file will contain a basic solution to Assignment 1, as well as "**vendors**" directory containing three **.json** files (**grand.json**, **indigo.json**, and **staples.json**). Each of these files contains the vendor data for one vendor 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 required directive.

The provided HTML file **orderform.html** is the same as for the previous assignment. In general, you should not change this file unless you need some additional elements to support your design decisions.

The JavaScript file **client.js** contains a sample solution to Assignment 1. It is well commented. You are encouraged to use your own solution to Assignment 1.

The JavaScript file **server.js** is empty. Use it to implement your server for providing resources to the client.

The Cascading Style Sheets file **styles.css** contains some basic styles for the web page. You must add more design rules to the file to make your page visually appealing and well-structured.

The two **.png** files can be used to attach click events to your webpage. Feel free to use other images.

Server Load (10 marks)

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

If you receive an error "access to XMLHttpRequest at localhost:3000 has been blocked by CORS policy", then add the following headers to your server responses:

```
response.setHeader("Access-Control-Allow-Origin", "*");
```

Home Page and Header (10 marks)

Your server must serve a home page in response to **GET** requests for the URL "**/**". At a 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:

- The home page
- The order form
- The vendor 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 vendors used to populate the drop-down menu must be requested from the server.
2. When a vendor is selected from the drop-down menu or the drop-down menu selection changes, the new vendor data must be requested from the server. The previously selected vendor's data must be removed.
3. The page must provide, at a minimum: a list of all office supply items for the currently selected vendor, the price of each of those items, its stock, 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 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.

Add New Item to Category (20 marks)

Provide an interface for users to add new supply items to the current vendor by choosing one of the categories and entering information about a new item. You can create a drop-down menu of existing categories (for the selected vendor) or provide another way for a user to select a category. Create several input fields for the following data input:

- new item's name
- new item's price
- new item's stock
- you can create a "description" input field or keep some default value for that.

Clearly label the purpose of each input field. Adding some "placeholders" (default values) to the fields is an excellent way to avoid unnecessary errors due to missed information.

Create the "Add New Item" button after pressing on which the newly created item with the input name, price, stock, and description must be added to the Supply List of the current vendor as the last item in the specified category. You should be able now to add this item to your cart.

The server will be responsible for adding a new item to the current vendor and providing the updated vendor data when requested.

Vendor Statistics Page (20 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 vendor:

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 (10 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 the 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.

Academic Integrity

Submitting not original work for marks is a serious offence. We use special software to detect plagiarism. The consequences of plagiarism vary from grade penalties to expulsion, based on the severity of the offence.

You may:

- Discuss general approaches with course staff and your classmates,
- Show your web page, but not your code,
- Use a search engine / the internet to look up basic HTML, CSS, and JavaScript syntax.

You may not:

- Send or otherwise share your solution code or code snippets with classmates,
- Use code not written by you,
- Use a search engine / the internet to look up approaches to the assignment,
- Use code from previous iterations of the course unless it was solely written by you,
- Use the internet to find source code or videos that give solutions to the assignment.

Submit Your Work

To submit your assignment, you must **zip** all your files and submit them to the Assignment 2 submission on Brightspace. Do not submit the **node_modules** directory; otherwise, you will receive a **deduction of 10 points**. Name your .zip file "YourName-a2.zip". Make sure you download your .zip file and check its contents after submitting it. If your .zip file is missing files or corrupt, you will lose marks.

Your .zip file should contain all the resources required for your assignment to run. The TA should be able to start your **server.js** server and interact with your app after typing **http://localhost:3000** in a browser without any additional setup.