Shopping list

The "shopping list" exercise is in two parts.

First I provide the server and you have to code the client (HTML/CSS/JavaScript) allowing to maintain a shopping list offline from several devices.

In a second step, you will be able to do the server part responding to your client, but also to those proposed to ensure that your server is compliant with the requests.

In download on DVO you will find :

- ListOfRace-Sources.zip : contains the doc of the API to set up and the sources of the client and server programs in Delphi
- TODOList-VueJS.zip : the source of a TODO list management using VueJS for its display. You can use it as inspiration if you want.
- ListeCoursesServeur-Win32.exe : a local 32 bits Windows server to manage the shopping list without Internet connection.
- ListeCoursesServeur-Win64.exe: a local Windows 64 bits server to manage the shopping list without Internet connection.
- ListeCourses-Win32.msix: a shopping list update client for Windows in 32 bits.
- ListeCourses-Win64.msix: a shopping list update client for Windows in 64 bits.
- ListeCourses-Mac.zip : a shopping list update client for Mac (version for Intel processor but functional on M1 or M2).

The objectives of this TD are:

- Make you manipulate JavaScript on a real project and possibly a display framework (VueJS, ReactJS, jQuery, or whichever you want).
- Make you manipulate recent browser APIs.
- Make you think about a data synchronization technique.
- Make you work on an API server.

Test servers

To create your shopping list client you need a server.

Under Windows you can use the one that can be downloaded in 32 or 64 bits.

You just have to launch the program and authorize the firewall in "private network" (never "public network") if Windows asks you to. The local server offers its API at http://localhost :8073

If you are on Mac or Linux I can compile a server for your version but it may not work for different reasons. It is better if you work with the version available on the Internet.

The official server for this shopping list offers its API at https://esilv.olfsoftware.fr/td5/

What is a shopping list?

The shopping list contains products and quantities for each product.

You should be able to add a new product to the list and change the quantities of each.

Data synchronization

There are multiple methods to synchronize databases. The simplest one is to synchronize the history of changes and apply them on each device. This is what I propose here.

Every product addition and quantity change must be stored so that it can be transmitted to the server from time to time or on demand.

In the case of this exercise I recommend a sync button, which is easier to set up and test.

In “public" projects, it is preferable to hide this notion of synchronization by automating the sending of pending information from a clock or when a reconnection to the Internet is detected following an outage. Some JavaScript APIs of browsers give us the information. This is the approach chosen by the editors of online office suites (Google Docs, Microsoft Office 365 for example).

During the synchronization you send your changes and you also ask for updates made by other applications since your last synchronization.

Depending on the type of synchronization and the type of database, this step must be done in a particular direction. In our case this should not change anything.

The complete and up-to-date shopping list is provided only once: at the first connection to the server. After that only updates are exchanged.

You will see that it is not too complicated to implement.

Unfortunately this technique does not apply to all types of data such as when working on databases with many tables and autoincremented keys since several users can create things in the same tables offline with the server. This can generate anomalies during synchronization.

Validation of your customer program

If you work locally you will need to be able to switch online to validate your application.

At the end of the work on the first part everyone should be able to synchronize with the same server and get the same list even if updates are made somewhere.

At the end of the work on the second part, all client programs must be able to synchronize with each other's servers. Changing the synchronization server resets the list of processed data.

Part 1: Viewing and editing the shopping list from a web browser

To start this tutorial you will only handle files on the client side. Consider that you are on a static site, without a server. You don't need a database or a local web server.

Download the examples and ZIPs you need and set up your working folder.

Choose a framework: ReactJS, VueJS , Bootstrap, TailwindCSS or another if you prefer. You can also use the "AJAX JavaScript Test" examples or do everything by hand.

Start with an example or a blank HTML5 page template.

Put in the code the HTML part you will need for the display and update of the shopping list (product name and quantity).

Then add the JavaScript part that allows you to switch from the list to the entry of a new product or the modification of the quantity of an existing product.

The user must be able to close the browser or disconnect from the Internet and then retrieve their data when they return to your page (as a file or web address). So make sure that this information persists between launches of your browser and consider an option to clean up everything on demand.

Finally, process the API to retrieve the starting shopping list from a server and send the changes to it.

Once you have successfully synchronized your data with the server, try with multiple browsers open to ensure that your changes synchronize correctly. You can also use the compiled client programs provided for your tests.

Second part: development of an API server

Once you have finished developing the client part of your application, start developing a server that will respond to its API.

In this case you will have to reuse local web servers (XAMPP, WAMP, MAMP or EasyPHP) in order to work in PHP. You can work natively or use a framework if you prefer (but frankly, for three URL calls, it is really not necessary).

Of course the API does not change. It's up to you to change the URL of your shopping list management and point to your own server or to your neighbor's.

You are free to use or not a database on the server. It is recommended because it is likely to be accessed by several people, but management in the form of files is also possible. It's up to you to think about it and make your choices.

Ideally you should be able to put your server online on the hosting I provided you (or another one of your own) and share the URL of your server so we can test it.

Corrections and solutions

The correction will be put online before the next class.

It contains:

- source code for a PHP API server that uses a MySQL database to store information (shopping list, customer list and change log)
- the source code of a web client using VueJS and XMLHttpRequest for Ajax

In addition to the ZIP to download on DVO you will also be able to access (on request) the private GitHub repository which will allow you to follow the evolutions.

As always if you want me to look at and comment on what you have coded send me a ZIP of your files by email.